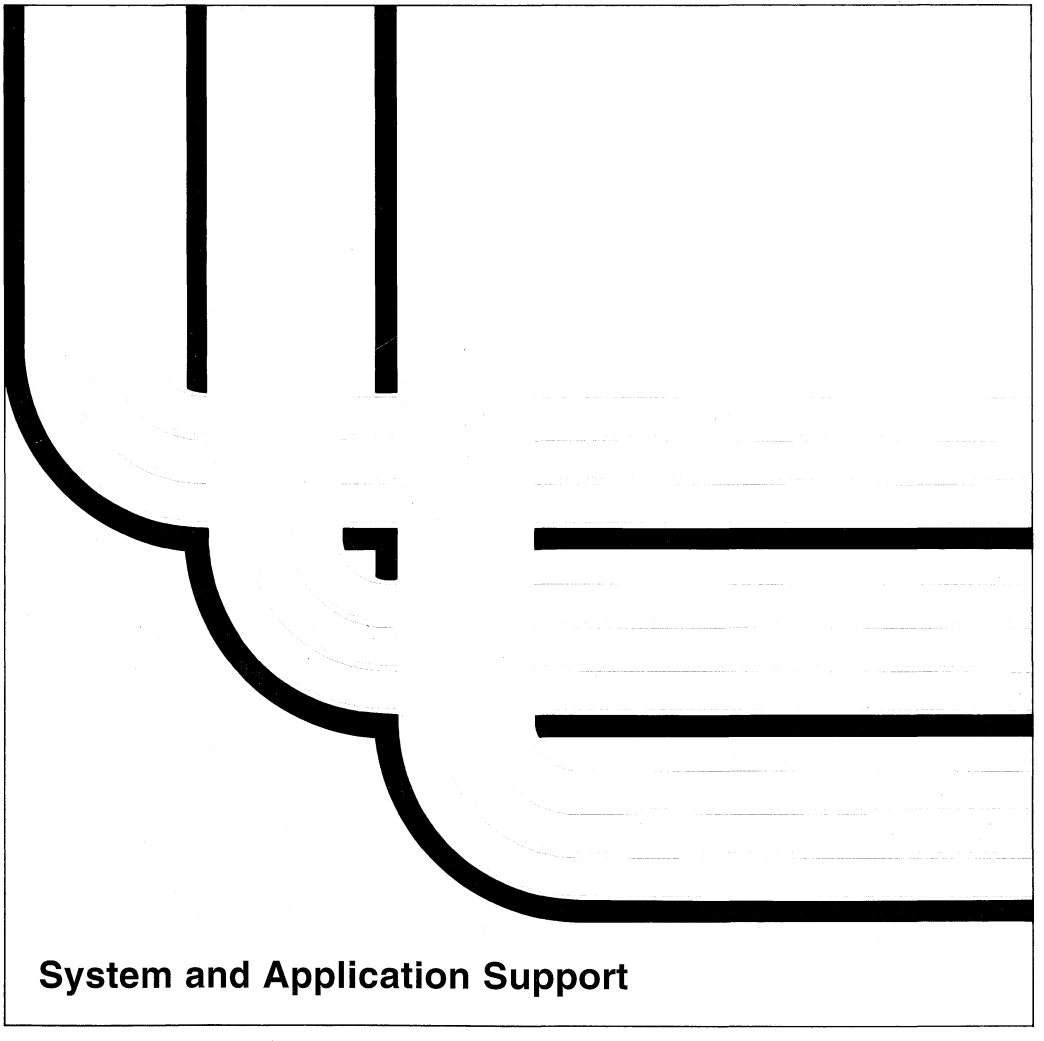
Application System/400

SC41-0030-02

Programming:
Control Language Reference
MONxxx through WRKxxx Commands
Appendixes

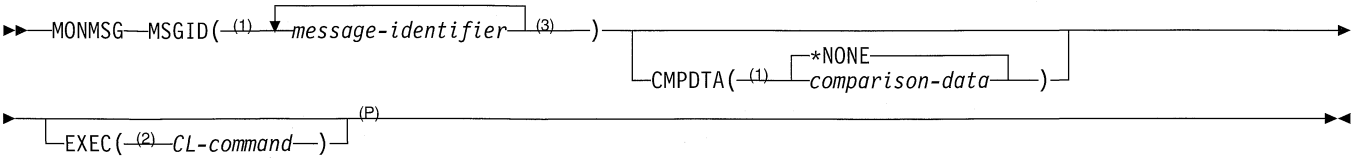


System and Application Support

(3)

MONMSG (Monitor Message) Command

Pgm: B,I



Notes:

1 A variable cannot be coded on this parameter.

2 If this MONMSG command is specified immediately after the PGM command or the last DCL command, only the GOTO command is valid here.

3 A maximum of 50 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Monitor Message (MONMSG) command is used to monitor escape, notify, and status messages sent to the program message queue of the program in which the command is being used. Completion and diagnostic messages cannot be monitored.

When the MONMSG command is compiled in a control language (CL) program, it establishes a monitor for the arrival of the specified messages. The command monitors the messages for the condition specified by the comparison data given in the command. If a message meeting the conditions arrives on the message queue, the CL command specified on the MONMSG command is processed.

Up to 1000 MONMSG commands can be specified in a program to monitor the arrival of messages for specific conditions or for a group of conditions. Specific message identifiers or generic message identifiers can be monitored. More information on the escape, notify, and status messages, and their identifiers, that can be sent by the CL commands, is in the *Programming Reference Summary*.

The MONMSG command can be coded following most commands in a CL program. A MONMSG command that is not placed at the beginning of the program applies only to the immediately preceding command; this is called a command-level MONMSG command. The command-level MONMSG command monitors only messages sent by the previous command. If the message sent by that command meets the conditions specified in the MONMSG command, the action specified in the same MONMSG command is taken. As many as 100 MONMSG commands, coded immediately after a command, can monitor the messages sent by that command.

When the action specified in the MONMSG command has been performed, and that action does not end with a GOTO or RETURN command, control returns to the command in the program that follows the command that sent the message. If the action ends with a GOTO command, control branches to the command in the program specified in the GOTO command. If the action ends with a RETURN command, control returns to the program that called the program that contains the MONMSG command.

If one or more MONMSG commands are placed at the beginning of the program, immediately following the declare commands or the PGM command if there are no declare commands, they monitor messages sent by all of the commands in the program (maximum of 100). This is called a program-level MONMSG command. If any message sent by any command in the program meets the conditions specified in any one of the program-level MONMSG commands, the corresponding action specified in the same command is taken.

The action taken by a command-level MONMSG command overrides a program-level MONMSG command.

If a command is coded for the EXEC parameter on a MONMSG command that is placed at the beginning of a program, *only* the GOTO command can be used, and it must specify the label for the command to which control is to be passed if a monitored message occurs. If a command is not coded for the EXEC parameter, monitored messages are ignored.

Restrictions:

1. This command is valid only in CL programs.
2. It can be coded after the last declare command (if declare commands are used), following the PGM command that begins the program, or it can be coded following any command allowed in CL programs, except for the following: DO, ELSE, ENDDO, ENDPGM, GOTO, IF, or RETURN. Note that if another program sends a message that is monitored by this command, a return cannot be made to that program.

Required Parameters

MSGID

Specifies the message identifiers of one or more escape, notify, or status messages that are monitored by this command. As many as 50 specific and/or generic message identifiers can be specified on one command.

Note: Many CL commands issue one escape message for many different error conditions. Details about the error or failure are given in diagnostic messages that precede the escape message. Although diagnostic messages cannot be monitored, they can be received from the job's external message queue after the escape message has activated the user's message monitor.

The first 3 characters must be a code consisting of an alphabetic character followed by 2 alphanumeric (alphabetic or decimal) characters; the last 4 characters can consist of the decimal numbers 0 through 9 and the characters A through F.

Note: Message identifiers using the MCH code (MCHnnnn) use only the numbers 0 through 9 in the last four characters.

If zeros are specified in either two or all four of the rightmost positions, such as ppmm00, a generic message identifier is specified. For example, if CPF0000 is specified, all the CPF messages are monitored.

Specify the message identifiers of 1 to 50 messages that are monitored when they arrive at this program's message queue. The identifiers of the escape, notify, and status messages that can be sent by the CL commands are in the *Programming Reference Summary*. CL variables cannot be used to specify message identifiers.

Optional Parameters

CMPDTA

Specifies the comparison data used to determine whether the monitored message (having one of the specified message identifiers) received on the program's message queue is acted on by this command. The message data specified in the MSGDTA parameter of the Send Program Message (SNDPGMMSG) command is compared with this comparison data. If the first part (up through the first 28 characters, or less) of the message's substitution values matches the comparison data specified, the action specified in the EXEC parameter of this command is taken. The action is also taken if no comparison data is specified.

***NONE:** No comparison data is specified; if the message in the program's message queue is from a command that this command is monitoring, and if it has the specified identifier, the action specified by EXEC is taken.

comparison-data: Specify a character string of no more than 28 characters, enclosed in apostrophes if necessary, that is compared with the same number of characters in the message data of the received message, starting with the first character in the message data. If the comparison data matches the first part of the received message data, this command performs the function specified in the EXEC parameter. A CL variable cannot be specified for the comparison data.

The comparison data can be displayed by the Display Program Variable (DSPPGMVAR) command.

EXEC

Specifies the CL command that is processed when a monitored message sent to the program's message queue meets the conditions specified in this MONMSG command. If no command is specified and a monitored message arrives on the queue, the message is ignored, and control passes to the next command in the program.

If the MONMSG command is placed at the beginning of the program, the EXEC parameter must specify the GOTO command and the label identifying the command that receives control.

Specify the CL command, including its parameters to be used, that is run when a message meeting the conditions specified in this command is received. The command specified is not run if the received message does not meet the specified conditions. A CL variable cannot be specified in place of the CL command.

Note: If a DO command is specified on EXEC, the entire DO group associated with the DO command is run if the condition is met.

Examples

Example 1: Monitoring Messages Sent by any Command

```
PGM
  MONMSG   MSGID(CPF0001  CPF1999)  EXEC(GOTO  EXIT2)
```

This example shows a MONMSG command at the beginning of a CL program that monitors for messages CPF0001 and CPF1999; these messages might be sent by any command processed later in the program. When either message is received from any of the commands running in the program, control branches in the program to the command identified by the label EXIT2.

CPF0001 states that an error was found in the command that is identified in the message itself. CPF1999, which can be sent by many of the debugging commands (like CHGPGMVAR), states that errors occurred on the command, but it does not identify the command in the message.

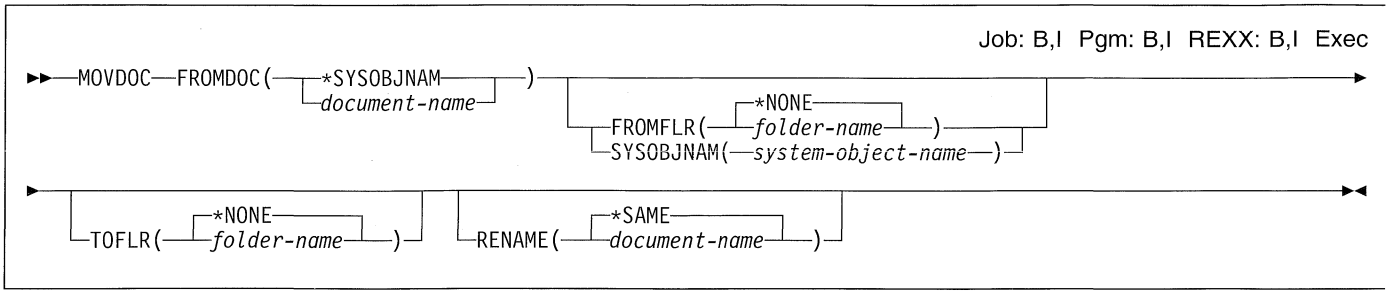
Example 2: Monitoring Messages Sent by a Single Command

```
CHGVAR  VAR(&A)  VALUE(&A / &B)
  MONMSG  MSGID(MCH1211)  EXEC(CHGVAR  VAR(&A)  VALUE(1)
```

In this example, the MONMSG command follows a Change Variable (CHGVAR) command and, therefore, is only monitoring messages sent by the CHGVAR command. The MI

escape message MCH1211 is sent to this program's message queue when a division by zero is attempted. Because MSGID(MCH1211) is specified, the MONMSG command is monitoring for this condition; when it receives the message, the second CHGVAR command is processed. In this command, the variable &A is set to a value of 1.

MOVDOC (Move Document) Command



Purpose

The Move Document (MOVDOC) command moves a document from one folder to another, removes a document from a folder, makes a document folderless, or moves a folderless document into a folder.

Restrictions: The user of this command must be enrolled in the system directory and have *ALL authority to the document. Also, if users are moving a document to a folder, they must have *CHANGE authority to the folder; and, if users are moving a document from a folder, they must have *CHANGE authority to the folder.

Required Parameters

FROMDOC

Specifies the document to be moved. If a document name is specified on the FROMDOC parameter, then a folder name must be specified on the FROMFLR parameter. If *SYSOBJNAM is specified on the FROMDOC parameter, then a system object name must be specified on the SYSOBJNAM parameter.

***SYSOBJNAM:** A system object name is used to identify the document being moved. This parameter must be specified when moving a folderless document, and it may be specified instead of a document name when moving a document from a folder. When *SYSOBJNAM is specified on the FROMDOC parameter, a user-defined variable must be specified on the SYSOBJNAM parameter and *document-name* must be specified.

document-name: Specify the name of the document that is moved.

Optional Parameters

FROMFLR

Specifies the folder from which the document is to be moved. A folder name must be specified on this parameter if a document name is specified on the FROMDOC parameter. *NONE cannot be specified on the FROMFLR parameter if document name is specified on the FROMDOC parameter.

***NONE:** The document to be moved is specified by its system object name.

folder-name: Specify the name of the folder from which the document is to be moved.

SYSOBJNAM

Specifies the system object name. This parameter is valid only when DLO(*SYSOBJNAM) or DOCL(*SYSOBJNAM) is specified. A full ten characters must be specified.

TOFLR

Specifies the name of the folder into which the document is to be moved. A folder name must be specified on this parameter if a document name is specified on the RENAME parameter. If the user specifies *NONE on the TOFLR parameter, the document becomes a folderless document and can only be referred to by its system object name.

***NONE:** The document becomes a folderless document and can only be referred to by its system object name.

folder-name: Specify the name of the folder to which the document is to be moved.

RENAME

Specifies a new name for the document in the TOFLR folder. This parameter allows the user to name a document when moving a folderless document to a folder. It also allows the user to rename the document when moving it from one folder to another.

If the user wants to move a document into a folder, the name of the document in the TOFLR folder must be unique.

If the new name is already assigned to a folder or a document in a folder specified on the TOFLR parameter, the user must either choose a new name for the target document or rename the folder or document that has the same name.

If the user specifies *SYSOBJNAM on the FROMDOC parameter, then a document name must be specified on the RENAME parameter.

***SAME:** The value does not change.

document-name: Specify the new name of the document in the TOFLR folder.

Examples

Example 1: Adding a Folderless Document

```
MOVDOC    FROMDOC(*SYSOBJNAM)  FROMFLR(*NONE)  TOFLR(FLR1)
   RENAME(DOC1)  SYSOBJNAM(CNTR192366)
```

This command, whose system object name is CNTR192366, adds a folderless document to FLR1 and names it DOC1.

Example 2: Moving a Document and Keeping its Name

```
MOVDOC    FROMDOC(DOC1)  FROMFLR(FLR1)  TOFLR(FLR2)
   RENAME(*SAME)
```

This command moves DOC1 from FLR1 to FLR2 and keeps the name DOC1.

Example 3: Moving and Renaming a Document

```
MOVDOC    FROMDOC(DOC1)  FROMFLR(FLR1)  TOFLR(FLR2)
   RENAME(DOC2)
```

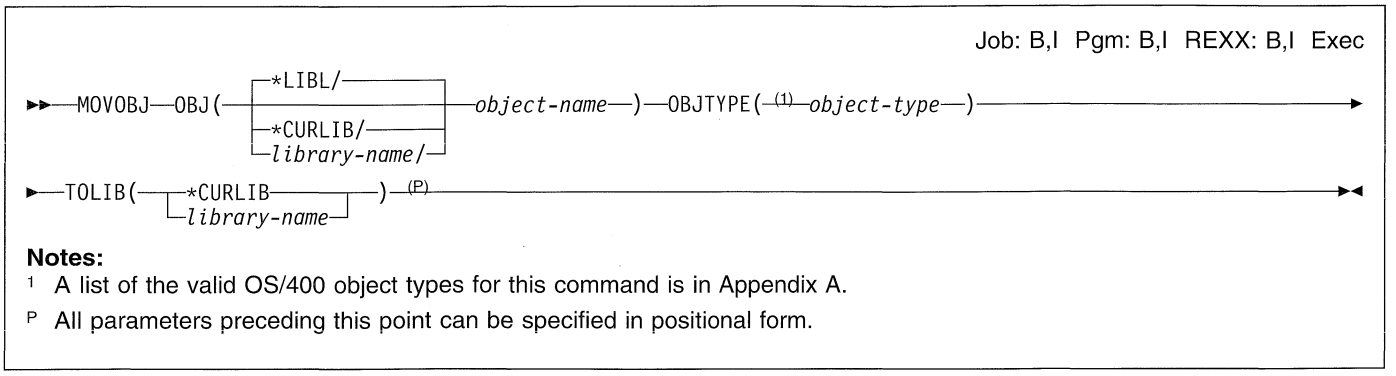
This command moves DOC1 from FLR1 to FLR2 and renames it DOC2.

Example 4: Moving a Document and Making It Folderless

```
MOVDOC    FROMDOC(DOC1)  FROMFLR(FLR1)  TOFLR(*NONE)
```

This command moves DOC1 from FLR1 and changes it to a folderless document.

MOVOBJ (Move Object) Command



Notes:
1 A list of the valid OS/400 object types for this command is in Appendix A.
P All parameters preceding this point can be specified in positional form.

Purpose

The Move Object (MOVOBJ) command removes an object from its currently assigned library and places it in a different library. The type of the object moved is specified in the OBJTYPE parameter.

Restrictions:

1. The user who submits this command must have object management authority for the object moved, delete and read authority to the library the object is currently in, and add and read authority to the library to which the object is being moved.
2. Libraries, user profiles, edit descriptions, line descriptions, control unit descriptions, device descriptions, journals, and journal receivers cannot be moved.
3. The following objects cannot be moved: the system operator message queue QSYSOPR, all work station user message queues, and the system log QHST.
4. The library to which the object is being moved must not already contain an object of the same name and type as the object being moved.
5. The user space (*USRSPC), user index (*USRIDX), and user queue (*USRQ) user domain objects can be copied only into libraries that are permitted in the system value QALWUSRDMN (allow user objects in library). However, if the user object was created in the system domain, it is not restricted.
6. Objects cannot be moved to the to library if the object and the to library are in different auxiliary storage pools (ASPs). An error message is issued when the object cannot be moved. You can move save files that are in a user ASP to libraries that are in the system ASP.

Required Parameters

OBJ

Specifies the qualified name of the object being moved to another library. (If no library qualifier is given, *LIBL is used to find the object.) The object name should be qualified to ensure that the correct object is moved.

The name of the object can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

object-name: Specify the name of the object that is moved.

OBJTYPE

Specifies the type of the object moved to another library. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: Journal and journal receiver objects are moved only from library QRCL to the library where they were originally allocated. An attempt to move objects of these types from or to any other library generates an error message.

TOLIB

Specifies the name of the library to which the object is moved. Library QTEMP cannot be specified.

***CURLIB:** The object is moved to the current library. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to which the object is moved.

Examples

Example 1: Moving an Object from the General Purpose Library

```
MOVOBJ  OBJ(QGPL/X)  OBJTYPE(*PGM)  TOLIB(MY)
```

The general purpose library is searched for the program (*PGM) object X. Before X is moved to MY library, the user profile of the user submitting the command is checked for (1) object operational and management authority for the object,

(2) add and read authority for MY library, and (3) delete and read authority for the library from which the object X is moved. After this command is run, object X is no longer in the QGPL library.

Example 2: Moving an Object from a Library in the Library List

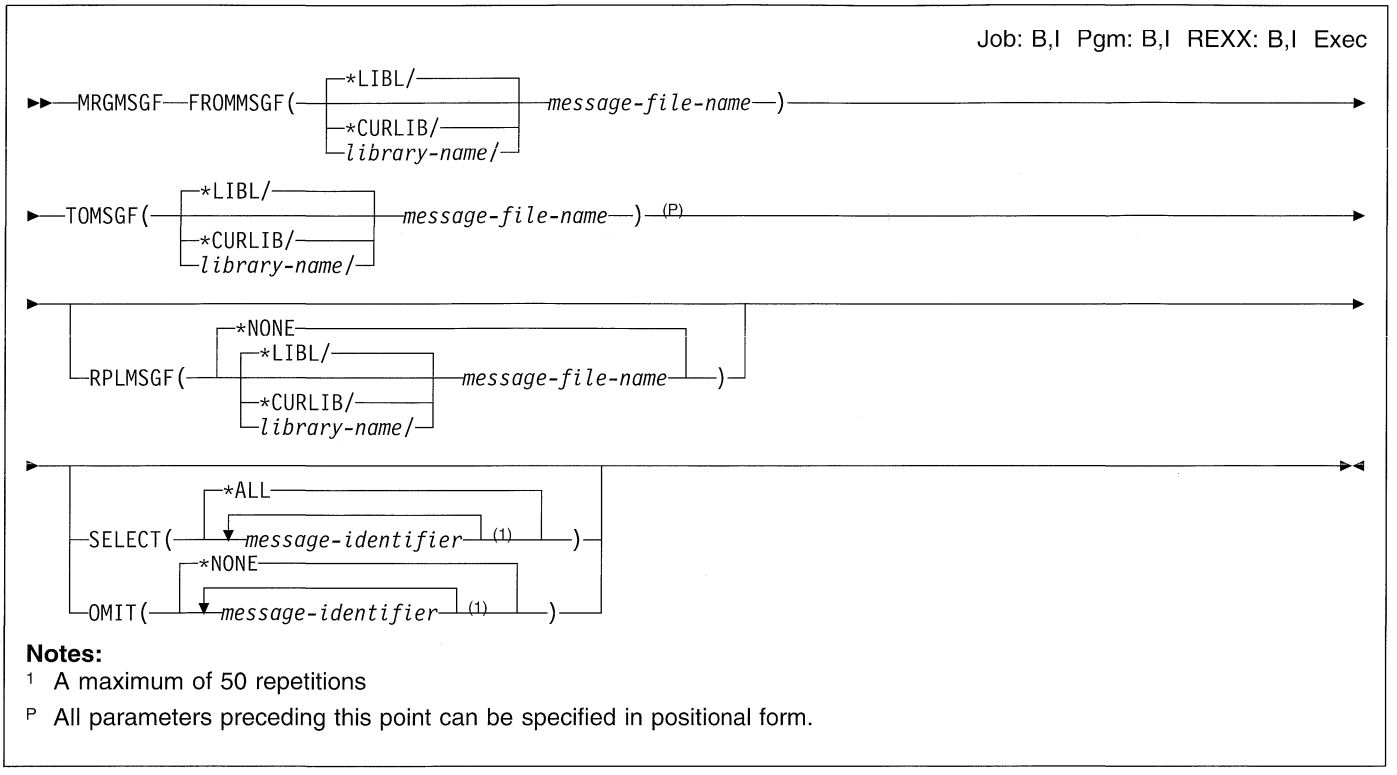
```
MOVOBJ  OBJ(*LIBL/Y)  OBJTYPE(*FILE)  TOLIB(Z)
```

or

```
MOVOBJ  Y  *FILE  Z
```

The library list (*LIBL) is used to locate the file named Y. If more than one object with the same name exists in the libraries making up the library list, the first object found via the library list for which the user has object operational authority is moved to library Z.

MRGMSGF (Merge Message File) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Merge Message File (MRGMSGF) command allows the user to merge messages from one message file with those in another message file. Another message file may be specified to hold the messages that are replaced during the merging process. None of the message files specified are deleted by this command (MRGMSGF).

Before the command is processed, messages can be in the from-message file (FROMMSGF), in the to-message file (TOMSGF), or in both files, but not in the replaced-message file (RPLMSGF). The three possibilities result in the following when the MRGMSGF command is processed:

- When the messages are only in the FROMMSGF, they are added to the TOMSGF
- When the messages are only in the TOMSGF, they remain in the TOMSGF
- When the messages are in both the FROMMSGF and the TOMSGF, the messages in the TOMSGF are first saved into the RPLMSGF (if a replace-message file is specified); then the messages in the TOMSGF are replaced by the messages in the FROMMSGF

Restriction: The user must have *USE authority to the from-message file (FROMMSGF); use, add, and delete authorities to the to-message file (TOMSGF); and use and add authorities to the replace-message file (RPLMSGF).

Required Parameters

FROMMSGF
Specifies the qualified name of the message file from which the messages are being merged.

The name of the from-message file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-file-name: Specify the name of the message file to use.

TOMSGF
Specifies the qualified name of the message file into which the messages from the FROMMSGF are being merged.

The name of the to-message file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-file-name: Specify the name of the message file to use.

Optional Parameters

RPLMSGF

Specifies the qualified name of the message file that will receive replaced messages (before the messages are replaced) from the to-message file (TOMSGF parameter) when the to-message and from-message files are merged with the merge message file (MRGMSGF) command.

***NONE:** A message file is not used to receive replaced messages.

The name of the message-file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-file-name: Specify the name of the message file to use.

SELECT

Specifies a list of the IDs of messages to be merged into the to-message file (TOMSGF parameter) from the from-message file (FROMMSGF parameter). All other messages in the from-message file are not merged.

Note: This parameter cannot be specified if the OMIT parameter is specified.

***ALL:** All messages in the from-file are merged into the to-file.

message-identifier: Specify up to 50 IDs of the messages to be merged.

OMIT

Specifies a list of the IDs of messages that are not merged into the to-message file (TOMSGF parameter) from the from-message file (FROMMSGF parameter). All other message IDs in the from-message file are merged.

Note: This parameter cannot be specified if the SELECT parameter is specified.

***NONE:** All messages in the from-file are merged into the to-file.

message-identifier: Specify up to 50 IDs of messages that are not being merged.

Examples

The examples below show how the merge files look before and after the MRGMSGF command is run.

Example 1: Merging Two Files

In the following example, two files are merged.

```
MRGMSGF FROMMSGF(A)  TOMSGF(B)
```

Table 56. Contents of the Files Before the Merge

Message File A	Message File B
ABC1234 'text A4'	ABC1233 'text B3'
ABC1236 'text A6'	ABC1234 'text B4'
ABC1237 'text A7'	ABC1235 'text B5'
ABC1238 'text A8'	ABC1236 'text B6'

Below are the two message files after the MRGMSGF command is processed. Notice that messages ABC1234 and ABC1236 are in both files. When the merge occurs, the message text from file A (text A4 and A6 respectively) replaces the message text in file B (text B4 and B6 respectively).

Table 57. Contents of the Files After the Merge

Message File A	Message File B
ABC1234 'text A4'	ABC1233 'text B3'
ABC1236 'text A6'	ABC1234 'text A4'
ABC1237 'text A7'	ABC1235 'text B5'
ABC1238 'text A8'	ABC1236 'text A6'
	ABC1237 'text A7'
	ABC1238 'text A8'

Example 2: Merging Two Files with Replace File Option

In the example below, messages that are replaced in the to-file are saved to a separate file before being replaced.

```
MRGMSGF  FROMMSGF(A)  TOMSGF(B)  RPLMSGF(C)
```

Table 58. Contents of the Files Before the Merge

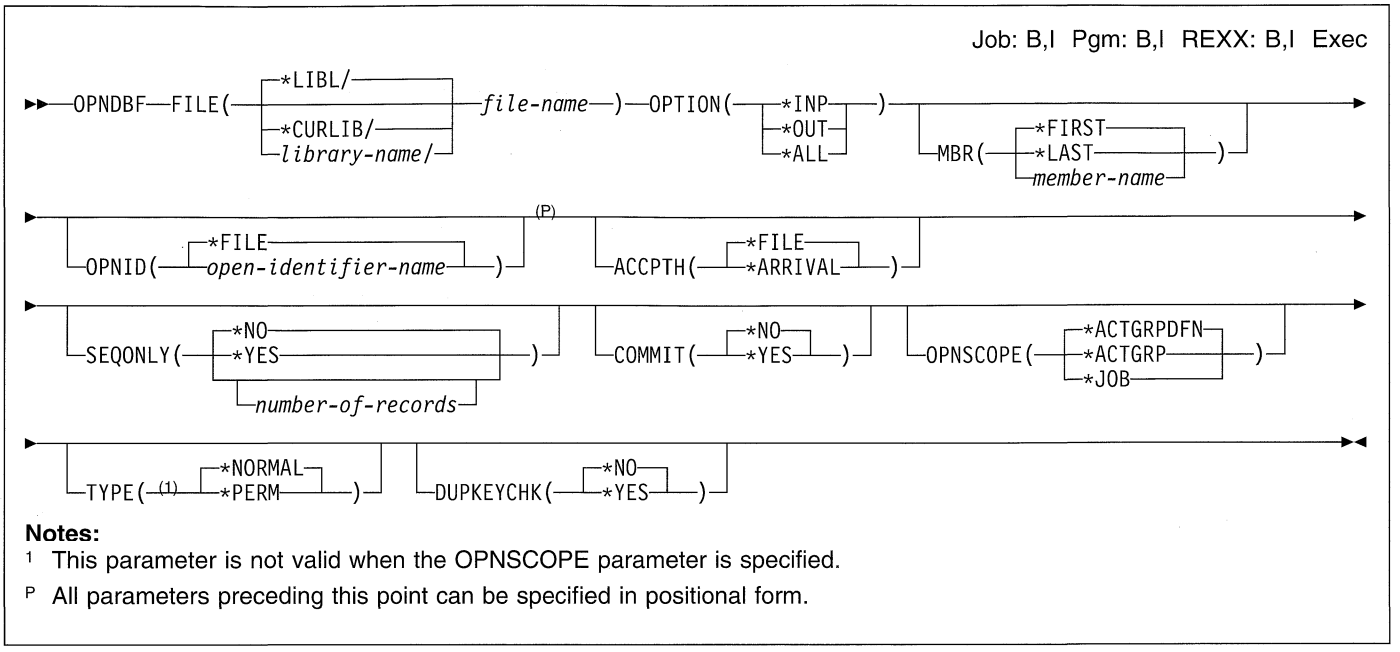
Message File A	Message File B	Message File C
ABC1234 'text A4'	ABC1233 'text B3'	
ABC1236 'text A6'	ABC1234 'text B4'	
ABC1237 'text A7'	ABC1235 'text B5'	
ABC1238 'text A8'	ABC1236 'text B6'	

Below are the two message files after the MRGMSGF command is processed. Notice that messages ABC1234 and ABC1236 are in both files. When the merge occurs, the text from these two messages is first moved to file C (text B4 and B6 respectively). Then, message text from file A (text A4 and A6 respectively) replaces the message text in file B (text B4 and B6 respectively).

Table 59. Contents of the Files After the Merge

Message File A	Message File B	Message File C
ABC1234 'text A4'	ABC1233 'text B3'	ABC1234 'text B4'
ABC1236 'text A6'	ABC1234 'text A4'	ABC1236 'text B6'
ABC1237 'text A7'	ABC1235 'text B5'	
ABC1238 'text A8'	ABC1236 'text A6'	
	ABC1237 'text A7'	
	ABC1238 'text A8'	

OPNDBF (Open Database File) Command



Notes:

1 This parameter is not valid when the OPNSCOPE parameter is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Open Database File (OPNDBF) command opens a database file member. Processing of records is done later by application programs that do shared open operations. If the file is opened by the OPNDBF command, this open operation is not used by the Receive File (RCVF) command. The RCVF command performs its own open operation. More information on database files is in the *Database Guide*.

Note: The share attribute of the open operation is determined by the current attributes of the file and any overrides currently in effect. A shared open operation means that subsequent open operations for which SHARE(*YES) is specified share this open data path (ODP), while subsequent open operations for which SHARE(*NO) is specified create their own ODP. If an override specifying a SHARE attribute is in effect when this command is sent, this attribute takes precedence over what is currently specified for the file; any other attribute specified on an Override Database File (OVRDBF) command also takes precedence over what is specified on this command.

Restrictions:

1. The user must have object operational authority to the database file, and add, update, delete, or read authority, or some combination of this authority on the physical files containing the data to be accessed. If the data is accessed by opening the physical file, the user must have object operational authority to the physical file. If the data is accessed by opening a logical file over the physical file, object operational authority to the logical file is required; object operational authority to the physical file is not required.
2. Subsequent shared open operations must adhere to the same open options (such as SEQONLY) that are specified when the OPNDBF command occurs.

Required Parameters

FILE

Specifies the qualified name of the file that contains the member being opened. Overrides currently in effect are processed.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the file that contains the member being opened.

OPTION

Specifies the options used to open a file. The options chosen on the first full open operation of a file are not changed on subsequent shared options.

***INP:** The file is opened for input operations only.

***OUT:** The file is opened for output operations only.

***ALL:** The file is opened for all options (input, output, update, and delete).

Optional Parameters

MBR

Specifies the member that is opened in the database file.

***FIRST:** The first member in the database file is used.

***LAST:** The last member of the specified physical file is opened.

member-name: Specify the name of the member that is opened.

OPNID

Specifies the identifier used for naming this open operation so it can be referred to when the member is closed or when the Position Database File (POSDBF) command is used. This identifier must be specified on the Close File (CLOF) command, and cannot be used on another OPNDBF command until the file is closed, or an escape message is sent and the open operation fails.

***FILE:** The file name is used for the open operation identifier.

open-identifier-name: Specify the name of this open operation.

ACCPTH

Specifies the access path to use for this open operation.

***FILE:** The file access path is used. If the file is keyed, the keyed access path is used; otherwise, the arrival sequence path is used.

***ARRIVAL:** The arrival sequence access path is used. If the file is keyed, the keyed access path is ignored. More information on arrival sequence processing is in the *Database Guide*.

SEQONLY

Specifies, for database files whose records are normally processed in sequential order, whether sequential-only processing is used on the file. This parameter also specifies the number of records transferred as a group to or from the database file if sequential-only processing is used. If an override specifying SEQONLY is in effect, it takes precedence over what is specified on this parameter. More information on sequential-only processing is in the SEQONLY parameter description on the Override with Database File (OVRDBF) command, and in the *Database Guide*.

***NO:** The database file does not use sequential-only processing.

***YES:** The database file uses sequential-only processing.

number-of-records: The file uses sequential-only processing. This parameter value indicates the number of records the database blocks up in its internal buffer before accessing the data in the member. Specifying this number is not required. If this value is not specified, the database chooses a default value. If this command specifies OPTION (*ALL) or COMMIT(*YES), the database changes the open operation to SEQONLY(*NO).

COMMIT

Specifies whether the SQL statements are run under commitment control.

***NO:** This file is not placed under commitment control.

***YES:** This file is placed under commitment control.

OPNSCOPE

Specifies the extent of influence (scope) of the open operation.

***ACTGRPDFN:** The scope of the open operation is determined by the activation group of the program that called the OVRDBF command processing program. If the activation group is the default activation group, the scope is the call level of the caller. If the activation group is a non-default activation group, the scope is the activation group of the caller.

***ACTGRP:** The scope of the open data path (ODP) is the activation group. Only those shared opens from the same activation group can share this ODP. This ODP is not reclaimed until the activation group is deactivated, or until the Close File (CLOF) command closes the activation group.

***JOB:** The scope of the open is the job in which the open occurs.

TYPE

Specifies the recursion level at which the reclaim resources function (RCLRSC) is effective.

***NORMAL:** The reclaim resources function is allowed to close the file if the program ends without doing a close operation.

***PERM:** The file remains open until it is closed by using the Close File (CLOF) command, or until the job ends. The open data path (ODP) remains in existence even if RCLRSC is used.

DUPKEYCHK

Specifies whether duplicate key feedback should be returned on I/O operations. Duplicate key feedback should only be requested for those files that are processed by COBOL programs since this is the only high-level language that uses the feedback. Duplicate key feedback should be requested only when it will be used by the COBOL application, since providing it can cause a performance degradation. More information on duplicate key feedback is in the *Database Guide*.

***NO:** Duplicate key feedback is not returned on I/O operations.

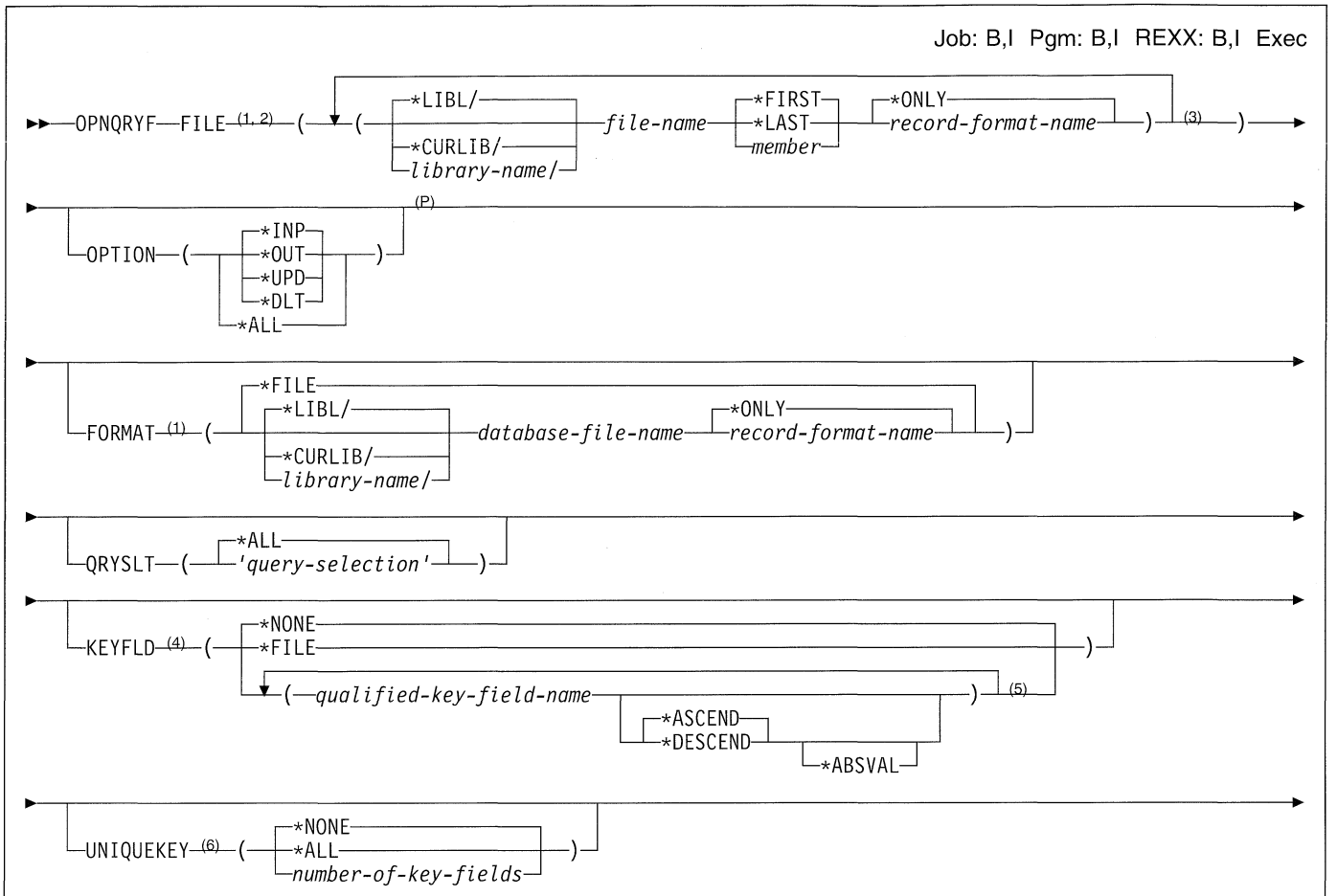
***YES:** Duplicate key feedback is returned on I/O operations.

Example

```
OPNDBF  FILE(MASTER/PAYROLL)  OPTIONS(*INP)
```

This command opens the first member in the file PAYROLL for input processing. The open identifier associated with this open operation has the file name as its identifier. If the file is specified as SHARE(*YES), subsequent open operations of the file PAYROLL (such as in an application program) perform more efficiently and use the same ODP.

OPNQRYF (Open Query File) Command



Notes:

1 When more than one FILE (file, library, member, record format) is specified, FORMAT(*FILE) cannot be specified.

2 When more than one FILE is specified, JFLD(*NONE) is allowed only when the default is specified for JDFTVAL and JORDER.

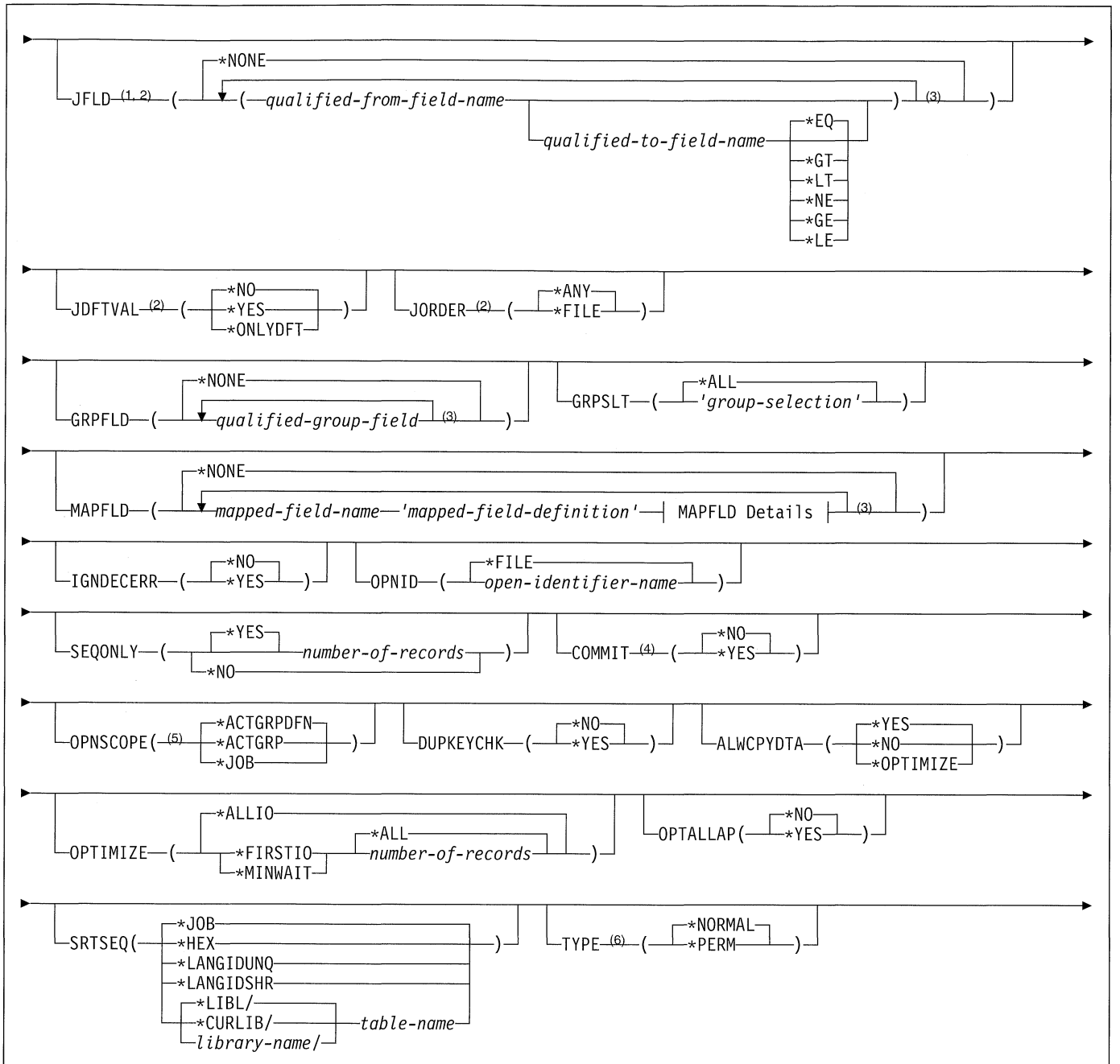
3 A maximum of 32 repetitions

4 KEYFLD(*FILE) is not allowed whenever a value other than the default is specified on the GRPFLD or GRPSLT parameters.

5 A maximum of 50 repetitions

6 UNIQUEKEY(*NONE) is required whenever KEYFLD(*NONE) is specified.

P All parameters preceding this point can be specified in positional form.



Notes:

1 When more than one FILE (file, library, member, record format) is specified, FORMAT(*FILE) cannot be used.

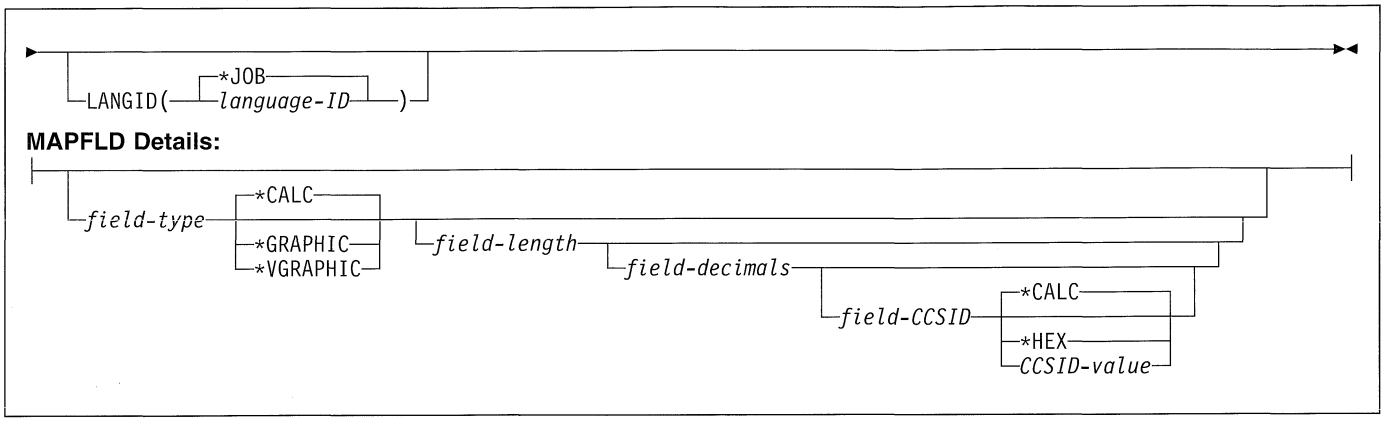
2 When more than one FILE is specified, JFLD(*NONE) is allowed only when the default is specified for JDFTVAL and JORDER.

3 A maximum of 50 repetitions

4 COMMIT(*NO) is required whenever a value other than the default is specified on the GRPFLD or GRPSLT parameters.

5 This parameter is not valid when the TYPE parameter is specified.

6 This parameter is not valid when the OPNSCOPE parameter is specified.



Purpose

The Open Query File (OPNQRYF) command opens a file that contains a set of database records that satisfies a database query request. Once opened, the file looks like a database file opened by using the Open Database File (OPNDBF) command, and the records in the file are accessed by high-level language programs that share the open data path (ODP). The path is closed, and all query resources are deallocated, by using the Close File (CLOF) command.

This command is used to do any combination of the following database functions:

- Join records from more than one file, member, and record format. The join operation that is performed may be equal or non-equal in nature.
- Calculate new field values by using numeric and character operations on field values and constants.
- Group records by like values of one or more fields, and calculate aggregate functions, such as minimum field value and average field value, for each group.
- Select a subset of the available records. Selection can be done both before and after grouping the records.
- Arrange result records by the value of one or more key fields.

More Command Functions: Following the parameter descriptions and the command examples (on pages P3-1652 through P3-1654) is the "Additional Considerations" section. Included in it are subsections on:

- Field Names - on page P3-1655
- Expressions - on page P3-1655
- Built-In Functions - on page P3-1657
- Restricted Built-In Functions - on page P3-1665

Restrictions:

1. The user can use overrides to change the file, library, and member names specified on the FILE parameter. Overrides are ignored for the file and library specified on the FORMAT parameter, unless FORMAT(*FILE) is specified. Parameter values specified on an override command, other than TOFILE, MBR, LVLCHK, WAITRCD, SEQONLY, or INHWRT and SHARE, are ignored by the OPNQRYF command.
2. The OPNQRYF command does not share an existing open data path (ODP) in the job or activation group. If an existing SHARE(*YES) ODP in the job or activation group has the same file, library, and member name as the open query file open data path (ODP), the query file does not open and an escape message is sent.
3. Each subsequent shared open operation must use the same open options (such as SEQONLY) that are in effect when the OPNQRYF command is run.
4. Some system functions (such as the Display Physical File Member (DSPPFM) and Copy File (CPYF) commands) do not share an existing open data path. The OPNQRYF command cannot be used with those functions.
5. The file opened with the OPNQRYF command cannot be used in programs written in BASIC because BASIC does not share an existing open data path.
6. Users of this command must have the following authorities:
   - Read authority for any library that is needed to locate the files specified on the FILE and FORMAT parameters
   - Object operational authority for any physical or logical file specified on the FILE parameter, and one or more of the following data authorities for the physical file or based-on physical file members of a logical file member:
     - Read authority if the file is opened for input (using option *INP)
     - Add authority if the file is opened for output (using option *OUT)
     - Update authority if the file is opened for updates (using option *UPD)
     - Delete authority if the file is opened for deletions (using option *DLT)
     - Read, add, update, and delete authority if the file is opened for all I/O operations (using option *ALL)
   - Object operational authority for any file specified on the FORMAT parameter
   - Use authority for any translate tables specified on the MAPFLD parameter (using option *USE)

Notes:

1. The Copy from Query File (CPYFRMQRYF) command can be used to copy data from a data path opened with the OPNQRYF command.
2. More information about the OPNQRYF command is in the *Database Guide*.

Required Parameter

FILE

Specifies one or more files, members, and record formats that are processed by the open query file command. All files specified must be physical files, logical database files, or Distributed Data Management (DDM) files. If DDM files are used, all files they refer to must be on the same target system, and the target system must be an IBM AS/400 system or IBM System/38.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the files, members, and record formats that are processed.

Element 1: Member Values

***FIRST:** The oldest member created is used.

***LAST:** The newest member created is used.

member: Specify the file member name.

Element 2: Record Format Values

***ONLY:** Only the record format in the file is used. If no record format name is specified, *ONLY is the default. If the file has more than one record format, a record format name is specified.

record-format-name: Specify the name of the record format that is used. The record format must exist in the database file specified in the first part of the FILE parameter.

When more than one file, file member, and record format is specified, the query joins field values to create a single set of records. Any file specified in the list may be a physical, join logical, or SQL view. More information on SQL views is in the *SQL/400* Reference*. The maximum number of physical file members that can be joined by a single query is 32. This limit includes the based-on physical file members for any join logical file member or SQL view member specified on the FILE parameter. The limit also includes each separate occurrence of the same physical file member when it is specified more than once in the list, either directly by name or by being referred to through a logical file member.

Optional Parameters

OPTION

Specifies the open option used for the query file. The options chosen on the first full open operation of a file are not changed on subsequent shared open operations. The user can specify either OPTION(*ALL) or up to four of the following values in any order:

***INP:** Open the file for input. OPTION(*INP) is the only value allowed if join processing or group processing is requested, if UNIQUEKEY processing is specified, or if all the fields in the open query file record format (specified on the FORMAT parameter) are for input-only use.

***OUT:** Open the file for output. In some high-level languages, output to certain files (such as files defined as 'direct access' in the high-level language program) is done by using a combination of input and updates. OPTION(*UPD) or OPTION(*ALL) is specified to use an open query file with such a program.

***UPD:** Open the file for update operations. If an input operation comes before an update, *INP on the OPTION parameter must be specified when OPTION(*UPD) is specified.

***DLT:** Open the file for delete operations. If each delete operation is preceded by an input operation, *INP on the OPTION parameter must be specified when OPTION(*DLT) is specified.

Other Single Values

***ALL:** Opens the file for all operations (*INP, *OUT, *UPD, and *DLT).

FORMAT

Specifies the record format used for records made available by using the OPNQRYF command. The simple field names in the open query file record format must represent fields that are either defined on the MAPFLD parameter or are unique across all files, members, and record formats specified on the FILE parameter. The value for any field that has the same name as a field specified on the MAPFLD parameter is determined by the mapped-field definition on the MAPFLD parameter. The value for any field not defined on the MAPFLD parameter is determined by a mapping of the field with the same name in one of the based-on files, file members, and record formats specified on the FILE parameter. Only the name, type, length, decimal positions, keyboard shift, and usage attributes of each field specified in the record format that is identified on the FORMAT parameter are used for processing the OPNQRYF command. All other attributes are ignored. The attributes do not have to be the same. If they differ, the fields are mapped in a way similar to those described for the Change Variable (CHGVAR) command.

***FILE:** The record format of the first or only entry on the FILE parameter is used. FORMAT(*FILE) is not allowed when more than one file, member, and record format are specified on the FILE parameter (therefore requiring a join query).

Element 1: File Name

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of a physical or logical database file, or a Distributed Data Management (DDM) file that contains the record format that is used.

Element 2: Record Format Values

***ONLY:** The only record format in the file is used. If no record format name is specified, *ONLY is the default. If the file has more than one record format, specification of a record format name is required.

record-format-name: Specify the name of the record format that is used. The record format must exist in the database file specified in the first part of the FORMAT parameter.

QRYSLT

Specifies the selection values used (before grouping) to determine which records are available by using the open query file.

***ALL:** All records in the physical or logical files, members, and record formats specified on the FILE parameter (after join processing, if required) are selected.

'query-selection': Specify an expression of up to 2000 characters (enclosed in apostrophes) that describes the values used to determine which records are selected. Any logical expression formed from relations can be specified (such as *EQ and *NE) of field and constant values or functions of field and constant values. At least one field name is specified in each relation. However, a field cannot be specified that depends on an aggregate function either directly in its definition or indirectly by referring to a mapped field. Refer to the "Expressions" section of "Additional Considerations" on page P3-1655 of this command description for a complete list of the operators used for this parameter.

For example, to select all records for which the value of field CUSNBR is less than 7000 and the value of field BALDUE is greater than 90% of the value of field CRLIMIT, specify the following:

```
QRYSLT('CUSNBR<7000 *AND
BALDUE>CRLIMIT*0.9')
```

Each field name may be qualified with either a file name or number that indicates which element in the list of files, members, and record formats specified on the FILE parameter contains the field. The special value *MAPFLD may be used to qualify the field name if the field is defined on the MAPFLD parameter.

KEYFLD

Specifies the name of one or more key fields used to arrange the query records, or specifies that the access path sequence of the first or only file, file member, and record format specified on the FILE parameter is used to arrange the query records. If key field names are specified, the user must also indicate whether the part of the key associated with each key field is ascending or descending, and whether the records are arranged by the absolute value of a numeric key field. If the qualified key field specified is a double-byte character field, the data is arranged in a single-byte sequence.

All key fields for an open query file must appear in the query records processed through the file. The fields named in the record format identified on the FORMAT parameter must include all key fields for the open query file, even if KEYFLD(*FILE) is specified to use the existing access path of a keyed file.

***NONE:** Key fields are not used to arrange the query records; therefore, any arrangement is acceptable. It is possible for the system to display query records in different arrangements if the same query is run twice, based on such factors as the current number of records in the file members being queried. KEYFLD(*NONE) allows the system more flexibility than other KEYFLD values to improve the performance record processing through the open query file.

***FILE:** The query records have the same arrangement as the first file, file member, and record format specified on the FILE parameter. KEYFLD(*FILE) is specified even if the first file in the list has only an arrival sequence access path, in which case the query record arrangement matches the arrival sequence of the first file, file member, and record format specified on the FILE parameter.

When KEYFLD(*FILE) is specified and a sort sequence other than *HEX has been specified on the SRTSEQ parameter, you may receive your records in an order that does not reflect the true file order. If this is a key file, the query's sort sequence is applied to the key fields of the file and an informational message is sent. If the file has a sort sequence table or an alternative collating sequence table, it is ignored for the ordering. This allows users to indicate which fields to apply a sort sequence to without having to list all the field names. If a sort sequence is not specified for the query, the query is ordered as in releases previous to V2R3M0.

Element 1: Qualified Key Field Values

qualified-key-field-name: Specify one or more field names (up to 50 field names can be entered) used to

define a keyed access path to arrange the query records. Each field name is qualified with either a file name or number that indicates which element in the list of files, members, and record formats specified on the FILE parameter contains the field. The special value *MAPFLD is also used to qualify the field name if the field is defined on the MAPFLD parameter.

Each field name on the KEYFLD parameter must be a field name that appears in the query records defined by the FORMAT parameter. The sum of the lengths of all key fields cannot be more than 10,000 bytes. In addition, if the sum of the lengths of the key fields is greater than 2000 bytes, *INP must be specified on the OPTION parameter and fields cannot be ordered by their absolute value.

Element 2: Key Field Order

***ASCEND:** The part of the key defined by the specified key field is ordered by ascending key values.

***DESCEND:** The part of the key defined by the specified key field is ordered by descending key values.

Element 3: Order by Absolute Value

***ABSVAL:** The part of the key defined by the specified key field is arranged by the absolute value of the key field. *ABSVAL is specified together with either *ASCEND or *DESCEND, but it is ignored if the key field is not numeric. If *ABSVAL is not specified, the records are arranged by the signed value of a numeric key field.

UNIQUEKEY

Specifies whether the query is restricted to records with unique key values, and specifies how many of the key fields must be unique. If *ALL or *number-of-key-fields* is specified, null values are considered to be equal.

***NONE:** None of the key fields specified on the KEYFLD parameter must be unique. All query records are available through the open query file, regardless of key value.

***ALL:** All key fields specified on the KEYFLD parameter must be unique. If there are multiple query records with the same values for all of the key fields, only the first such record is available through the open query file.

number-of-key-fields: Specify the number of key fields that must be unique. This value must be no larger than the number of key fields determined by the KEYFLD parameter. If there are multiple query records with the same value for the specified number of consecutive key fields, only the first such record is available through the open query file.

JFLD

Specifies whether the query joins records from multiple file members, and specifies how to join field values from the files, members, and record formats specified on the FILE parameter in constructing the query records.

The first file, member, and record format specified on the FILE parameter is called the join primary, and all other elements specified on the FILE parameter are called join secondaries. The JFLD parameter specifies a list containing pairs of field names, where the first field in each pair provides a value that is used to select records in a join secondary that has the same value in the second field name of the pair.

The join from-field and to-field may be simple or mapped fields (specified on the MAPFLD parameter), but a field that depends on an aggregate function (either directly in its definition or indirectly by referring to a mapped field) cannot be used.

The join from-field and to-field are not required to have identical field attributes, but numeric fields cannot be mixed with character or double-byte character set (DBCS) fields, and character fields cannot be mixed with DBCS-only field types. If the fields do not have identical attributes, the system converts them to identical attributes for join processing. This change uses a packed decimal format if both fields are fixed-point numeric fields, or floating-point format if either field is of the floating-point type. The change for fixed-point numeric fields aligns the decimal points and pads with zeros. Numeric-type change truncates fractional digits if more than 31 total digits are required for fixed-point numbers, or drops some of the least significant digits if more than 15 total digits are required for floating-point numbers. Character and DBCS-open fields are changed by padding the shorter field with blanks. DBCS-either fields are padded with blanks if the field contains SBCS data or padded with double-byte blanks if the field contains DBCS data. DBCS-only fields are padded with double-byte blanks.

If more than one file is specified on the FILE parameter, and if the JDFTVAL(*NO) value and JORDER(*ANY) value are specified, the system takes information from the JFLD and QRYSLT parameters and derives the final join specifications. If a file is specified on the FILE parameter and the file is not referred to by the QRYSLT or JFLD parameters, all records for that file are logically joined to all other records created from the other files specified on the FILE parameter.

If either JDFTVAL(*YES), JDFTVAL(*ONLYDFT), or JORDER(*FILE) is specified, the join fields must be specified on the JFLD parameter.

***NONE:** No join operation is specified. If more than one file is specified on the FILE parameter, and JDFTVAL(*NO) and JORDER(*ANY) is specified, the system automatically finds the join fields from the QRYSLT parameter.

Element 1: From-Field Name

qualified-from-field-name: Specify a field name to provide the value used to select records in a join secondary file, file member, and record format. The field name is qualified with either a file name or number that indicates which element in the list of files, file members, and record formats (specified on the FILE parameter) contains the field. The special value *MAPFLD can also

be used to qualify the field name if the field is defined on the MAPFLD parameter.

A join from-field is either a simple field or a mapped field defined on the MAPFLD parameter. If JDFTVAL(*YES) or JDFTVAL(*ONLYDFT) is specified, it must depend only on fields contained in the join primary or in join secondaries specified on the FILE parameter ahead of the join secondary associated with the to-field of the pair.

Element 2: To-Field Name

qualified-to-field-name: Specify a field name used to select records from a join secondary file, file member, and record format in constructing the query records. The field name is qualified with either a file name or number that indicates which element in the list of files, file members, and record formats specified on the FILE parameter contains the field. The special value *MAPFLD is used to qualify the field name if the field is defined on the MAPFLD parameter.

A join to-field is either a simple field or a mapped field defined on the MAPFLD parameter. If JDFTVAL(*YES) or JDFTVAL(*ONLYDFT) is specified, it must depend only on fields contained in a single join secondary. If the join secondary is a join logical file, only fields contained in the primary physical file member for the join logical file are components of the join to-field. The sum of the lengths of all to-fields for each join secondary (after change, if the from-field and to-field attributes are not identical) cannot be more than 2000 bytes unless JDFTVAL(*NO) is specified. Up to 50 join field pairs are specified.

Element 3: Join Operators

join-operator: Specifies the type of join operation that is performed for the specified from-field and to-field. If JDFTVAL(*NO) and JORDER(*ANY) are specified, or if more than one join field pair is specified, a different join operator may be specified for each pair. If JDFTVAL(*YES), JDFTVAL(*ONLYDFT), or JORDER(*FILE) is specified, then only one join operator may be specified, regardless of the join pairs.

***EQ:** An equal join operation is performed.

***GT:** A greater than join operation is performed.

***LT:** A less than join operation is performed.

***NE:** A not equal join operation is performed.

***GE:** A greater than or equal join operation is performed.

***LE:** A less than or equal join operation is performed.

JDFTVAL

Specifies whether the query file includes join records that use default values for any of the fields. The default values are from a join secondary file that contains a record that does not match the corresponding record in the primary file. The matching attempted is the join criteria as specified on the JFLD parameter.

Join processing attempts to collect field values from the join primary and join secondaries. It does so by matching join from-field values to records in a join secondary that produce the appropriate values in the join to-field. If there are no records in a join secondary to produce the to-field values required for the pairs of join fields associated with the join secondary, this parameter specifies whether query records should be constructed by using default values for all fields obtained from the join secondary.

If the FILE parameter includes any join logical or SQL views, all join files must be compatible with the JDFTVAL value. If the data description specification (DDS) used to create a queried join logical file does not contain the JDFTVAL keyword, JDFTVAL(*NO) must be specified. If any join logical file specified on the FILE parameter has the JDFTVAL keyword, then all join logical files for this open query file must be created by using the JDFTVAL keyword, and JDFTVAL(*YES) must be used. If any files are SQL views, then JDFTVAL(*NO) is required.

If the JDFTVAL parameter is not compatible with the attributes of the join logical files or SQL views being processed, the join view files specified on the FILE parameter can be replaced with their based-on physical file members. The correct, additional from-field and to-field pairs can be provided on the JFLD parameter to join records from the physical file members in any way.

If more than one file is specified on the FILE parameter, and if either JDFTVAL(*YES) or JDFTVAL(*ONLYDFT) is specified, the system uses the join fields as specified on the JFLD parameter as the final join specifications.

***NO:** Default values are not used to construct join query records.

***YES:** Creates all records for the join operation, including those produced both with and without default values. If *YES is specified, no SQL views are allowed.

***ONLYDFT:** Creates only the records produced by using default values in constructing the join operation. This option is used to include only exception records in the records available through the open query file. If *ONLYDFT is specified, no join logical files or SQL views can be specified on the FILE parameter.

JORDER

Specifies, for a join query, whether the join order must match the order specified on the FILE parameter. If the join order is varied, the query records are created in a different arrangement. If the value specified for the JDFTVAL parameter is *YES or *ONLYDFT, the JORDER parameter is ignored. The order specified on the FILE parameter is always preserved, because changing the join order can change which records are returned when JDFTVAL processing is required.

If more than one file is specified in the FILE parameter and JORDER(*FILE) is specified, the system uses the

join fields as specified on the JFLD parameter as the final join specifications.

***ANY:** Any join file order is allowed, and any such arrangement may be used by the system to create the result records. It is possible for a query to return result records in a different arrangement if the same query is run twice, consecutively (based on factors such as the current number of records in the files being queried). JORDER(*ANY) allows the system more flexibility to improve the performance of processing records through the open query file than any other JORDER parameter value.

***FILE:** The order of the file, file member, and record format elements specified on the FILE parameter are preserved in the join operation.

GRPFLD

Specifies the field name or names used to group query results. One query record is created for each group of records (after join processing, if required) selected by the QRYSLT parameter. The group is defined by the collection of records that has the same set of values for the fields specified in the record format identified on the FORMAT parameter. If no field names are specified and group processing is required, the whole file is considered to be one group. Each query record that is created is either made available through the open query file or is discarded, depending on the selection values specified on the GRPSLT parameter. All null values within a grouping column are considered equal. To ensure an ascending sequence, the KEYFLD parameter must also be specified.

***NONE:** Fields are not used to form groups. If the grouping function is required (because selection values are specified on the GRPSLT parameter, or an aggregate function is used by a field specified on the MAPFLD parameter), records selected by the values specified on the QRYSLT parameter are handled as a single group.

qualified-group-field: Specify one or more field names (up to 50) to group the query results. Each field name may be qualified with either a file name or number to indicate which element in the list of files, members, and record formats specified on the FILE parameter contains the field. The special value *MAPFLD may also be used to qualify the field name if the field is defined on the MAPFLD parameter.

A grouping field defined on the MAPFLD parameter cannot refer to an aggregate function in its definition (either directly, or indirectly through the use of another field specified on the MAPFLD parameter). The sum of the lengths of all grouping fields cannot exceed 2000 bytes.

GRPSLT

Specifies the selection values used after grouping to determine which records are available through the open query file.

***ALL:** All records defined by the grouping function described in the GRPFLD parameter description are selected.

'group-selection': Specify an expression (up to 2000 characters enclosed in apostrophes) that describes the values used to determine which records are selected. Any logical expression formed from relations (such as *EQ and *NE) of field and constant values or functions of field and constant values are specified. Only grouping fields (specified on the GRPFLD parameter), constants, aggregate functions (such as %AVG and %STDDEV), and mapped fields (specified on the MAPFLD parameter) that are composed of grouping fields and aggregate functions can be referred to in any relation. At least one field must be specified in each relation. Refer to the "Expressions" section of "Additional Considerations" on page P3-1655 of this command description for a complete list of the operators used for this parameter.

For example, to select all records for which the value of grouping field OVRDUE is greater than or equal to 10, and the average value of field CREDIT in each group is less than 100, specify the following:

```
GRPSLT('OVRDUE>=10 *AND %AVG(CREDIT)<100')
```

Each field name may be qualified with either a file name or number that indicates which element in the list of files, file members, and record formats specified on the FILE parameter contains the field. The special value, *MAPFLD, may also be used to qualify the field name if the field is defined on the MAPFLD parameter.

MAPFLD

Specifies the definition of query fields that are either mapped to, or derived from, other fields. MAPFLD is generally not needed if the field names specified on other OPNQRYF command parameters are simple field names that exist in only one of the file, file member, and record format elements specified on the FILE parameter.

***NONE:** Mapped fields are not needed. All field names specified on other parameters exist in some record format specified on the FILE parameter.

Element 1: Mapped Field

mapped-field-name: Specify the simple field name used on any other OPNQRYF command parameter that must refer to this mapped field. A qualified name is *not* allowed for the first part of the MAPFLD parameter list item. All specified mapped-field-name values must be unique. Refer to the "Expressions" section of "Additional Considerations" on page P3-1655 of this command description for a complete list of the operators used for this parameter.

Element 2: Field Definition Expression

'mapped-field-definition': Specify an expression (up to 256 characters enclosed in apostrophes) that defines the mapped field in terms of other fields that either exist in only one of the file, file member, and record format ele-

ments specified on the FILE parameter, or are defined by some other mapped field definition appearing earlier in the MAPFLD list. Either numeric operations or string operations are allowed, depending on the data type of the fields used in the definition. Refer to the "Expressions" section of "Additional Considerations" on page P3-1655 of this command description for a complete list of the operators used for this parameter.

For example, to define a mapped field named WHSPAR as the concatenation of the field WHRSE with the first 10 characters of field PART, specify the following:

```
MAPFLD((WHSPAR 'WHRSE *CAT %SST(PART 1 10)'))
```

Each field name is qualified with either a file name or a number that indicates which element in the list of files, file members, and record formats specified on the FILE parameter contains the field. The special value *MAPFLD is also used to qualify the field name if the field is defined in an earlier list item on the MAPFLD parameter.

Element 3: Mapped Field Type

field-type: Specify the field type for this mapped field, or specify *CALC to allow the system to calculate appropriate attributes (including field type) for the mapped field. *CALC is the default if no field type value is specified.

When *CALC is used, the field attributes are determined in one of two ways. The attributes either match the field definition in the record format identified on the FORMAT parameter, or (if the field is not in the record format on the FORMAT parameter) the attributes are calculated based on the expression specified in the mapped-field definition for this field. If the mapped field is used in the record format identified on the FORMAT parameter, either use *CALC or specify attributes (field type, field length, and field decimals) identical to those of the field in the record format specified on the FORMAT parameter.

The field type must be valid for the final result of the expression specified on the mapped-field definition. Character and numeric types are only mixed if the numeric type is in zoned decimal format and the field length and the length of the expression result are the same. A character type may be mapped to a DBCS-open or DBCS-either type, but a DBCS type cannot be mapped to a character type.

The following are the only DBCS mappings allowed on the MAPFLD parameter:

- From character to DBCS-open type
- From character to DBCS-either type
- From DBCS-either to DBCS-either type
- From DBCS-open to DBCS-open type
- From DBCS-only to DBCS-only type
- From DBCS-only to DBCS-open type
- From DBCS-only to DBCS-either type
- From DBCS-only to DBCS-graphic type
- From DBCS-graphic to DBCS-only type
- From DBCS-graphic to DBCS-open type
- From DBCS-graphic to DBCS-either type
- From DBCS-graphic to DBCS-graphic type

The Query Field Structure table (see Table 60) shows the allowable *field type* and *external field length* values. It also shows the *default field length* and *decimal positions* and the *internal field length* (in bytes) for each type of field.

Element 4: Length

field-length: Specify the field length in number of digits for a numeric field, number of bytes for a character or DBCS field, or number of characters for a graphic field. A field length must be an even value for DBCS-only and DBCS-either field types. The range of valid lengths for each field type is shown in the Query Field Structure table. A field length value is specified if *CALC is used for the field type.

Element 5: Decimal Positions

field-decimals: Specify the number of decimal positions for a numeric field, expressed as a number of decimal digits, that is no larger than the total number of digits specified for the field length. If no value is given, the default value is assumed to be zero. A field decimal's value must not be specified for a binary or character field, or if *CALC is specified for the field type.

Element 6: CCSID Value

field-CCSID: Specify the CCSID (character code set identifier) being assigned to the mapped field being created. This parameter is valid only for fixed- or variable-length character, DBCS, or hexadecimal fields, or for *CALC fields that result in one of these types. If no value is specified, the CCSID is determined based on the CCSIDs of the fields and/or literal values specified on the MAPFLD definition.

Literal values in the MAPFLD definition are tagged with the job default CCSID. However, if the MAPFLD definition consists of only a literal value and the user specifies a field CCSID value, the literal will be tagged with that CCSID. This allows the user to tag a literal with a CCSID other than the job default CCSID.

Note: Normally, *HEX and *VHEX fields do not have an associated CCSID. Because of this, the data in the field is treated the same regardless of the default CCSID of the system that the data is being used on. However, if the user specifies a CCSID for a *HEX or *VHEX field, the CCSID overrides the hexadecimal attribute of the field (causing the field to be treated as *CHAR/*VCHAR), and the data in the field may be treated differently if it is moved to a system that has a different default CCSID.

For more information on CCSIDs, see the *Database Guide*.

An example of the MAPFLD parameter, showing the use of field-type, field-length, field-decimals, and field-CCSID, is as follows:

```
MAPFLD((UNTPRICE 'TOTPRICE / UNTCOUNT' *DEC  7 2)
  (SHORTSTR '%SST(LONGSTR 1 5)' *CHAR 5 *N 930))
```

IGNDECERR

Specifies whether the system ignores decimal data errors during query file processing.

***NO:** The system does not ignore decimal data errors.

***YES:** The system ignores decimal data errors. When errors in decimal data are encountered, the invalid sign and/or digits are automatically changed to valid values.

OPNID

Specifies the identifier used to name the query file so it is referred to by the Close File (CLOF) command when it is closed or by the Position Database File (POSDBF) command when it is opened. The identifier must differ from the OPNID associated with any other file which was previously opened by using the OPNDBF command or OPNQRYF command, and which is not yet closed.

***FILE:** The name of the first or only file specified on the FILE parameter is used for the open query file identifier.

open-identifier-name: Specify the name to associate with this open query file.

SEQONLY

Specifies whether sequential-only processing is used for the query file, and also specifies the number of records that can be processed as a group when read or write operations are performed on the open query file. The open query file open data path (ODP) uses a different SEQONLY value than the one specified on this parameter. It depends on other parameter values specified on the OPNQRYF command. A message is sent if the SEQONLY value is changed. More information about sequential-only processing is in the SEQONLY parameter description on the OVRDBF (Override Database File) command, and in the *Database Guide*.

Element 1: Sequential-Only Processing is Used

***YES:** The open query file uses sequential-only processing. Number of records can be specified with this value.

Element 2: Number of Records That can be Processed

number-of-records: Specify the number of records that are processed as a group when read or write operations are performed with the open query file. If no number-of-records value is specified, the system calculates the number of records processed as a group.

Other Single Values

***NO:** The file does not use sequential-only processing.

COMMIT

Specifies whether the SQL statements are run under commitment control.

***NO:** The open query file is not placed under commitment control.

***YES:** The open query file is placed under commitment control.

OPNSCOPE

Specifies the extent of influence (scope) of the open operation.

***ACTGRPDFN:** The scope of the open operation is determined by the activation group of the program that called the OVRDBF command processing program. If the activation group is the default activation group, the scope is the call level of the caller. If the activation group is a non-default activation group, the scope is the activation group of the caller.

***ACTGRP:** The scope of the open data path (ODP) is the activation group. Only those shared opens from the same activation group can share this ODP. This ODP is not reclaimed until the activation group is deactivated, or until the Close File (CLOF) command closes the activation group.

***JOB:** The scope of the open operation is the job in which the open operation occurs.

DUPKEYCHK

Specifies whether duplicate key feedback is returned on input/output (I/O) operations. Duplicate key feedback should be requested only for files that are processed by COBOL programs since this is the only high-level language (HLL) that utilizes the feedback. Duplicate key feedback should only be requested when it is used by the COBOL application since providing it can cause a performance degradation. More information on duplicate key feedback is in the *Database Guide*.

***NO:** Duplicate key feedback is not returned on I/O operations.

***YES:** Duplicate key feedback is returned on I/O operations.

ALWCPYDTA

Specifies whether the system is allowed to copy data from the files, file members, and record formats specified on the FILE parameter. If so, the system is allowed to open the query file to the copy. The system tries to avoid using a copy of the data because a copy does not reflect changes made to the database after the information is copied. However, certain requests (such as when key fields contained in multiple based-on files for a join are specified) require that the data be copied before the specified query functions are performed.

***YES:** The system may use a copy of data from the files, file members, and record formats specified on the FILE parameter. A copy of the data is used only when it is needed to perform the requested query functions.

***NO:** The system does not use a copy of data from the files, file members, and record formats specified on the FILE parameter. If it is necessary to use a copy of the

data to perform the requested query functions, the query file is not opened and an error message is sent.

***OPTIMIZE:** The system uses a sort routine to order the output from the files, file members, and record formats specified on the FILE parameter. A sort routine is used only if the KEYFLD parameter is specified, and if using a sort routine would improve query performance without conflicting with other OPNQRYF options.

A sort will improve the performance of a query that returns most or all of the records in the file or files specified on the FILE parameter.

Using a sort can increase the time required for the OPNQRYF command to process. This occurs because the sort is performed and all records to be returned through the query are processed while the OPNQRYF command is active. However, because the records are already processed, the reading of the records (by using either a program or the CPYFRMQRYF command) is very fast. Therefore, the overall time to process the query is reduced.

Specifying the KEYFLD parameter for the OPNQRYF command does not ensure that the query will use an index if ALWCPYDTA(*OPTIMIZE) is specified. If a sort routine is used, the file is not opened with indexed access. If the program reading the records from the OPNQRYF command requires indexed access (random processing rather than sequential processing), ALWCPYDTA(*YES) or ALWCPYDTA(*NO) should be specified.

When a sort is used, the query file's position is not changed when a ROLLBACK command is issued. Therefore, when a query is opened that has parameters, ROLLBACK commands that follow do not reset the queried file's position to where it was at the start of the unit of recovery.

Note: Do not specify ALWCPYDTA(*OPTIMIZE) if you require that a ROLLBACK command reposition the query file, or if you require that the queried file be opened with indexed access.

The following items are required before a sort is valid for the OPNQRYF command:

- ALWCPYDTA(*OPTIMIZE) must be specified.
- The OPTION parameter, if specified, must be *INP.
- A value other than *FILE or *NONE must be specified on the KEYFLD parameter.
- The UNIQUEKEY parameter must not be specified, or must specify *NONE.
- The SEQONLY parameter, if specified, must be *YES.
- If COMMIT(*YES) is specified, the level of record locking (LCKLVL parameter on the STRCMTCTL command) must not be *ALL.
- The DUPKEYCHK parameter must not be specified, or must specify *NO.
- The total buffer length of all fields in the file specified on the FORMAT parameter (or FILE parameter, if the FORMAT parameter is not specified) must not exceed 32700 bytes.

The query optimizer determines whether a sort is used. This decision is based on the number of records expected from the query and the options specified on the OPNQRYF statement. The following items influence the optimizer's choice of a sort:

- The OPTIMIZE parameter should specify *ALLIO or *MINWAIT. If *FIRSTIO is specified, the number of records specified should be close to or equal to the number of result records expected from the query.
- The number of records in a file specified on the FILE parameter should contain a minimum of 200 records.
- The query result should contain a minimum of 200 records.

OPTIMIZE

Specifies the most efficient way the system can perform the selection and join processing necessary to satisfy the other parameter specifications on the OPNQRYF command.

If the KEYFLD or GRPFLD parameters require that an access path be built (when no existing access path is shared), the access path is built completely, regardless of the OPTIMIZE entry. Optimization primarily affects system operation when selection processing is performed.

***ALLIO:** The system attempts to reduce the total input/output time required to process the whole query, assuming that all query records are read from the file.

Element 1: Time Reduction Options

***FIRSTIO:** The system attempts to reduce the input/output time required to open the query file and to retrieve the first buffer of records from the file.

***MINWAIT:** The system attempts to reduce the time required to open the query file by minimizing delays when records are read from the file.

Element 2: Number of Records to Retrieve

***ALL:** All the query records are used when the query is optimized.

number-of-records: Specify the number of query records to use in the optimize operation.

OPTALLAP

Specifies whether the query optimizer should consider all the access paths that exist over the files being queried when determining how to implement the query.

***NO:** Allow the query optimizer to operate normally. When determining how to implement a query, the optimizer considers access paths until an internal timeout value has been exceeded. If there are a large number

of access paths over the files being queried, the optimizer may time out before it has considered all the available access paths.

***YES:** Force the query optimizer to ignore the internal timeout value and consider all the available access paths over all the files in the query. Note that if there are a large number of access paths over the files it may take a long time to optimize the query.

SRTSEQ

Specifies the sort sequence to be used for sorting and grouping selections specified on the QRYSLT or GRPSLT parameters, joins specified on the JFLD parameter, ordering specified on the KEYFLD parameter, grouping specified on the GRPFLD parameter, %MIN or %MAX built in functions, or unique key values specified on the UNIQUEKEY parameter.

***JOB:** The SRTSEQ value for the job is retrieved for the job.

***HEX:** A sort sequence table is not used. The hexadecimal values of the characters are used to determine the sort sequence.

***LANGIDUNQ:** A unique-weight sort table is used.

***LANGIDSHR:** A shared-weight sort table is used.

The name of the sort sequence table can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

table-name: Specify the name of the sort sequence table to be used with this query.

LANGID

Specifies the language identifier to be used when SRTSEQ(*LANGIDUNQ) or SRTSEQ(*LANGIDSHR) is specified.

***JOB:** The LANGID value for the job is retrieved for the job.

language-ID: Specify the language identifier to be used by the job.

TYPE

Specifies the recursion level at which the Reclaim Resources (RCLRSC) command closes the file.

Note: This parameter is ignored unless the default value is specified on the OPNSCOPE parameter and the request is from the default activation group.

***NORMAL:** The RCLRSC command closes the file if the program call that ran the OPNQRYF command is ended without closing the file.

***PERM:** The file remains open until the Close File (CLOF) command closes it, or until the routing step or default activation group ends. The query file remains open even if the RCLRSC command is run.

Table 60. Query Field Structure

Type	Field Type	External Field Length	Default Length and Decimals	Internal Field Length in Bytes
Binary	*BIN2	1-5	5 0[2]	2
Binary	*BIN4	1-10	10 0[2]	4
Floating-point	*FLT4	1-9	7 6[1]	4
Floating-point	*FLT8	1-17	15 14[1]	8
Packed decimal	*DEC	1-31	15 5[1]	1-16
Zoned decimal	*ZONED	1-31	15 5[1]	1-31
Character	*CHAR	1-32766	32	1-32766
Variable length character	*VCHAR	0-32740	32	0-32740
Hex	*HEX	1-32766	32	1-32766
Variable length hex	*VHEX	0-32740	32	0-32740
DBCS-only	*ONLY	4-32766[3]	32	4-32766
DBCS-either	*EITHER	4-32766[3]	32	4-32766
DBCS-open	*OPEN	4-32766	32	4-32766
DBCS-graphic	*GRAPHIC	1-16383[5]	32	2-32766
Variable length DBCS-only	*VONLY	0-32740	32	0-32740
Variable length DBCS-either	*VEITHER	0-32740	32	0-32740
Variable length DBCS-open	*VOPEN	0-32740	32	0-32740
Variable length DBCS-graphic	*VGRAPHIC	0-16370[5]	32	0-32740
Date	*DATE	5-10[4]	8	4
Time	*TIME	4-8[4]	7	3
Timestamp	*TIMESTP	14; 16-26[4]	26	10

1 If the number of decimal digits is specified, but the decimal precision value is omitted for an *FLT4, *FLT8, *DEC, or *ZONED field, the precision defaults to zero decimal digits.
2 A nonzero value is not allowed for the decimal precision of a binary number.
3 The field length must be an even-numbered value.
4 These fields can be longer if they are padded on the right with blanks.
5 These field lengths are in a number of graphic characters.

Examples

Example 1: Selecting Specific Records

```
OPNQRYF   FILE(ordfile)  OPTION(*all)
  QRYSLT('orddate=%range("840101" "841231")
  & ordamt>100')
  KEYFLD((ordamt *descend))
```

This command uses the QRYSLT parameter to select only records in the first member of file ORDFILE that have an order date in 1984 and an order amount greater than 100. Because the FORMAT parameter is omitted, the open query file has the same record format as file ORDFILE. The open query file allows all file operations (input, output, update, and delete). The KEYFLD specification is used to force the records to be arranged by descending value of order amount.

Example 2: Using the %XLATE Built-In Function

```
OPNQRYF   FILE(telefile)
  QRYSLT('%xlate(usrname qsystrntbl) *ct "GEORGE"')
```

This command uses the %XLATE built-in function to translate the field USRNAME to uppercase, and to instruct the *CT operator to select only records that contain the value GEORGE in the field USRNAME. QSYSTRNTBL is an IBM-supplied system translation table that converts lowercase alphabetics (a through z) to uppercase (A through Z). The translation is done to ensure that the search value is recognized even if its characters appear in mixed case. The records available through the open query file have the same record format as those in file TELEFILE.

Example 3: Using the %XLATE Built-In Function

```
OPNQRYF   FILE(telefile)
  QRYSLT('usrname *ct ''GEORGE''')
  MAPFLD((usrname
  '%xlate(telefile/usrname qsystrntbl)'))
```

In the previous example, the value of field USRNAME, which is returned to the high-level language (HLL) program that reads records from the open query file, is not translated to uppercase.

This example shows a way to make the uppercase version of field USRNAME available to the HLL program. This is done by defining a mapped field (MAPFLD parameter) for the translated value of field USRNAME. The field has the same field name as the field name in the open query file record format being used. The translated version of the field is used for selection (QRYSLT parameter) and is used in the open query file record format.

Example 4: Using the %SST Built-In Function

```
OPNQRYF   FILE((histlib/ordfile hist1))
  OPTION(*inp *upd *dlt)
  FORMAT(ordinfo orddtls)
  QRYSLT('month=7')
  MAPFLD((year '%sst(orddate 1 2)' *zoned 2)
  (month '%sst(orddate 3 2)' *zoned 2)
  (day '%sst(orddate 5 2)' *zoned 2))
```

This command uses the %SST built-in function to create a substring of the year, month, and day parts of character field ORDDATE in file ORDFILE. The command also maps a character string to a zoned field, which is valid as long as the zoned field has the same length as the character string. If the file ORDINFO has a record format, ORDDTLS, containing at least the field's YEAR, MONTH, and DAY records, these fields have input-only usage in the open query file record format because they are defined by using a built-in function (%SST) and are mappings that mix character and numeric (zoned decimal format) types. The file is opened for input, update, and delete operations, but none of the field's YEAR, MONTH, and DAY records are updated using the open query file open data path (ODP). The open query file uses only records in the HIST1 member of file ORDFILE in library HISTLIB, and the records retrieved through the file have the same format as record format ORDDTLS in file ORDINFO. Only records pertaining to the month of July are processed through the open query file (QRYSLT parameter).

Example 5: Returning the First Record of Each Set

```
OPNQRYF   FILE((routelf *first locusr))
  QRYSLT('%sst(toloc 1 4) *eq "ROCH"')
  KEYFLD(fromusr fromloc tousr toloc)
  UNIQUEKEY(*all)
```

This command uses the KEYFLD and UNIQUEKEY parameters to return only the first record of each set of records in record format LOCUSR in the first member of file ROUTELF that have the same values for the fields FROMUSR, FROMLOC, TOUSR, and TOLOC. The query result is further restricted by selecting only records that have the value ROCH in the first four characters of field TOLOC. The records available through the open query file contain all of the fields in record format LOCUSR of file ROUTELF. If the file ROUTELF contains information about messages routed by an application, this example identifies all unique sender and receiver pairs in which the receiving location name begins with ROCH.

Example 6: Joining a File to Itself

```
OPNQRYF   FILE(partpf partpf)
  FORMAT(partjoin)
  JFLD((1/pnbr 2/pnbr *GE))
  MAPFLD((pnm1 '1/pname') (pnm2 '2/pname')
  (pnbr '1/pnbr'))
```

This example illustrates how a file is joined to itself, as well as how to use the MAPFLD parameter to rename fields in the based-on files. A greater than or join is performed using field PNBR as both the join from-field and the join to-field.

The format of file PARTJOIN is assumed to contain fields named PNBR, PNM1, and PNM2. The field name PNBR is valid in the query output record format because that field is defined on the MAPFLD parameter. If the record format in file PARTJOIN contains a field named PNAME, an error occurs because the field exists in both files specified on the FILE parameter, and is not the name of a field defined on the MAPFLD parameter. The mapped field definitions are field names, so the attributes of fields PNM1 and PNM2 match the attributes of field PNAME, and the attributes of field PNBR in the open query file records match field PNBR in file PARTPF. Further, when a file is joined to itself, it is always necessary to specify a file number name for any field that is defined in the based-on file.

Example 7: Renaming Fields in Based-On Files

The same query can also be specified as follows:

```
OPNQRYF   FILE(partpf partpf)
  FORMAT(partjoin)
  QRYSLT('1/pnbr *GE 2/pnbr')
  MAPFLD((pnm1 '1/pname') (pnm2 '2/pname')
  (pnbr '1/pnbr'))
```

Because more than one file is specified on the FILE parameter, and the default value is specified for the JDFTVAL and JORDER parameters, the system takes the join specifications from the values specified on the QRYSLT parameter.

Example 8: Selecting Master Records With No Detail Records

```
OPNQRYF   FILE(cusmas ordfil)
  FORMAT(cusmas)
  JFLD((cusnbr ordfil/cusnbr))
  JDFTVAL(*onlydft)
  MAPFLD((cusnbr 'cusmas/cusnbr'))
```

This command uses a join query to select only master records that have no associated detail records. The master file (CUSMAS) is joined (equal join) to the detail file (ORDFIL) by the customer number field that appears in both record formats. The customer number field name is the same in both record formats (CUSNBR). Because CUSNBR is the name of a field defined on the MAPFLD parameter, everywhere the simple field name CUSNBR is used, the mapped field version of the CUSNBR field in file CUSMAS is used (including the open query file record format, which matches the customer master file record format). The JDFTVAL parameter indicates that only records that are produced by using default values are available through the open query file. Every master record that has associated detail records (with the same value of the customer number field) is excluded, and every master record that has no associated detail records creates a result record.

Example 9: Identifying Detail Records With No Associated Master Record

```
OPNQRYF   FILE(ordfil cusmas)
  FORMAT(ordfil)
  JFLD((cusnbr cusmas/cusnbr))
  JDFTVAL(*onlydft)
  MAPFLD((cusnbr 'cusnbr ordfil/cusnbr'))
```

This change of the previous example (using the same files) shows how to identify all detail records with no associated master record (in this case, all orders with an unregistered customer number):

Example 10: Calculating Basic Statistics

```
OPNQRYF   FILE(scores)
  FORMAT(clsstats)
  GRPFLD(clsid)
  GRPSLT('clsavg<70 & clsmax-clsmin>30')
  MAPFLD((clscnt '%count')
         (clsavg '%avg(usrscore)')
         (clsmin '%min(usrscore)')
         (clsmax '%max(usrscore)'))
```

This command uses the grouping function to calculate basic statistics for each group of records in file SCORES that have the same value in the field CLSID. Assuming file CLSSTATS has a record format containing field CLSID and all fields specified on the MAPFLD parameter, each record available through the open query file contains the value of the grouping field (CLSID) as well as the number of records included in the group and the average, minimum, and maximum values of field USRSCORE in the group. Selection occurs after grouping, so that records are created for groups only when the average value of USRSCORE in the group is less than 70 and the difference between the maximum and minimum scores in the group is greater than 30.

Example 11: Selecting Records With a Specific Value

```
OPNQRYF   FILE(itmmast)
  QRYSLT('itmcode=%range(32 50) & itmtype="P"')
  ALWCPYDTA(*no)
  OPTIMIZE(*firstio)
  SEQONLY(*yes 10)
  TYPE(*perm)
```

This command selects from the first member of file ITMMAST only the records that have a value of field ITMCODE in the range from 32 through 50 and also have a value of field ITMTYPE equal to the letter P. The ALWCPYDTA parameter specifies that the open query file must never use a copy of the records in file ITMMAST. The OPTIMIZE and SEQONLY parameter values cause the system to attempt to improve processing for the open query file to minimize the time needed to retrieve the first buffer of ten records. This combination of parameter values is a good choice if the file is used with a high-level language interactive inquiry program that shares the open query file open data path (ODP) and shows ten records on each display screen. The open data path (OPD) for the open query file is 'permanent' (TYPE parameter), which means that it remains open either until the file is closed by using the Close File (CLOF) command or until the routing step ends.

Example 12: Tagging a Literal with a Specific CCSID

```
OPNQRYF   FILE(itmmast)
  QRYSLT('itmtype=pfield')
  MAPFLD((pfield 'P' *CHAR 1 *N 930))
```

This command selects from the first member of file ITMMAST only the records that have a value of field ITMTYPE equal to the letter 'P' in character set 930. The mapped field is created so that the literal 'P' can be tagged with a specific CCSID.

If a literal is not tagged with a specific CCSID, it is assigned the CCSID of the job running the query. Because of this, if an OPNQRYF statement is part of a CL program that is shared among systems with differing CCSIDs (in different countries, perhaps), a query that uses a literal in the selection specifications may not return the same results on all systems, even though the data in the files is the same. This happens because the internal representation of the literal may be different when the CL program is run in a job with a different CCSID. This representation then may not match the same records in the file. Note that the internal representation of the data in the file does not change based on the CCSID of the current job.

Tagging the literal with a specific CCSID avoids this problem. A literal tagged with a specific CCSID keeps the same internal representation on all systems. The CCSID that is used to tag the literal should be the same as the CCSID assigned to the field against which the literal is being compared.

Example 13: Using a Nonjoin Query

```
OPNQRYF   FILE((EMPLOYEE))  KEYFLD((NAME))
  ALWCPYDTA(*OPTIMIZE)
```

This command returns all of the records in the EMPLOYEE file.

Example 14: Using a Join Query

```
OPNQRYF   FILE((EMPLOYEE) (MANAGEMENT))
  FORMAT(EMPLOYEE)  KEYFLD((NAME))
  JFLD((1/EMPID 2/MEMPID))  ALWCPYDTA(*OPTIMIZE)
```

This command returns all of the records required by the join criteria.

Additional Considerations

The file, library, and file member names used by the open data path (ODP) are the same as the first file and file member names specified on the FILE parameter, unless an override forces the use of a different file or file member name. The record format name of the open query file is the same as that specified on the FORMAT parameter.

The OPNQRYF command always opens a file with an open data path (ODP) that is shared, as if SHARE(*YES) is specified for the file. If the file, library, or file member name speci-

fied in the HLL program differs from the name of the open query file, an override command must be used to specify the correct file, library, and member names to allow the high-level language program to share the open query file ODP. If the first, or the only, member queried has an attribute of SHARE(*NO), SHARE(*YES) must be specified in an override to enable an HLL program to share the query file ODP.

If the OPNQRYF is scoped to the job, any subsequent OPNQRYF can share the ODP whether scoped to an activation group or the job. If the OPNQRYF is scoped to an activation group, any subsequent OPNQRYF can share the ODP if it is also scoped to the same activation group.

The open query file ODP also has a LVLCHK attribute that matches the first, or only, member used for the query. Shared opens of an open query file are level checked unless the first queried member has an attribute of LVLCHK(*NO) or LVLCHK(*NO) is specified either in the program that opens the file or in an override which occurs before the shared open. The level number for the open query file record format is the same as the record format identified on the FORMAT parameter.

An expanded discussion of field names, expressions, and built-in functions, and restricted built-in functions used with parameters on the OPNQRYF command follows:

Field Names

The field name used as the first part of an element in the list specified on the MAPFLD parameter must be a simple name, and the field names in the record format identified on the FORMAT parameter are always treated as simple names. Any other field name specified on an OPNQRYF command parameter (QRYSLT, KEYFLD, JFLD, GRPFLD, GRPSLT, or the field-definition expression part of the MAPFLD parameter) is a qualified field name, specified as follows:

field-name

Specify a simple field name that identifies a field that is defined on the MAPFLD parameter, or with a field name that is unique among all field names from all record formats included in the list specified on the FILE parameter. This form is not allowed if there is no MAPFLD parameter definition for the specified field name and the FILE parameter includes more than one record format that contains a field with the specified name, even if the same file and record format is specified more than once in the list on the FILE parameter.

For example, AMOUNT is valid if the field named AMOUNT is defined on the MAPFLD parameter. It is also valid if AMOUNT is not defined on the MAPFLD parameter, as long as there is only one field named AMOUNT in any record format specified on the FILE parameter.

file-name/field-name

Specify a field name that is qualified with the simple name of the file specified on the FILE parameter whose record format contains the field, but only if the simple file name is unique among all file names specified on the FILE parameter. This form is not allowed if the same simple file name is specified more than once in the list specified for the FILE parameter, even if different library, member, or record format names are used.

For example, WHS01/PARTNBR is valid if there is a field named PARTNBR in the record format for file WHS01, and file name WHS01 is only specified once on the FILE parameter.

file-nbr/field-name

Specify a simple field name that is qualified with the number of the element in the FILE parameter list for the record format that contains the field. The file-nbr qualifier must be specified without leading zeros. This form is only required if the same simple file name is specified more than once in the list specified on the FILE parameter.

For example, 2/BALDUE is valid if the second file record format in the list specified on the FILE parameter contains a field named BALDUE.

***MAPFLD/field-name**

Specify a simple field name that is qualified with the special value *MAPFLD if the field is defined on the MAPFLD parameter. When the field is defined, this form has the same meaning as specifying the simple field name with no qualifier. If the field is not defined on the MAPFLD parameter, *MAPFLD cannot be specified.

For example, *MAPFLD/AVGBAL is valid if the AVGBAL field is specified as the first part of one of the mapped field list elements specified on the MAPFLD parameter.

Expressions

Expressions specified on the QRYSLT, GRPSLT, and MAPFLD parameters are similar to expressions specified on other CL command parameters. Logical, relational, numeric, and string operations are performed by using combinations of field values and constants. Symbolic and named operators are supported, as well as many built-in functions, and parentheses are used to control the order of evaluation.

There are also differences in the expressions specified on OPNQRYF parameters and on other CL command parameters. The following list summarizes the ways that expressions on the QRYSLT, GRPSLT, and MAPFLD parameters differ from normal CL expressions:

- The expression string must be enclosed in apostrophes if it contains embedded blanks or special characters
- The following differences affect numeric and string literals:
  - Character string constants are quoted by using apostrophes or double quotes
  - The leading and trailing zeros of a numeric constant are significant parts of its attributes
  - Floating-point constants (including the special values *INF and *NEGINF) are used in expressions

- The following differences contrast CL variables with database fields:
  - No prefixed ampersand (&) is used in database field names
  - Qualified field names are supported
  - No 'logical' field type exists for database fields
  - Many additional data types are supported for database fields
- The following CL operators are not supported on the OPNQRYF command:
  - *BCAT or v>
  - *TCAT or v<
- The following additional operators are supported beyond CL support:
  - // for remainder
  - ** for exponentiation
  - *CT for 'contains' (character scan)
  - *XOR or && for 'logical exclusive or'
- The following differences affect built-in function support:
  - The %SWITCH built-in function is not supported
  - Many additional built-in functions are supported
  - Nested built-in functions and expressions for built-in function arguments (such as '%LOG(%SIN(x))') generally are allowed
  - To support expressions as built-in function arguments, any argument that is a signed numeric value or an expression (for example, '%MIN(3 (-2) x (y+4))') must be enclosed in parentheses

The following table shows the priority of all operators that are used for expressions on the QRYSLT, GRPSLT, or MAPFLD parameters. Only operators listed for priorities 1 through 5, excluding the *NOT and ¬ operators, are allowed in an expression specified on the MAPFLD parameter:

Priority	Operators
1	+, - (when used for signed numeric values), *NOT, ¬
2	**
3	*, / ,// (a / must have a space before the / and/or after the /)
4	+, - (when used between two operands)
5	*CAT, vv
6	*GT, *LT, *EQ, *GE, *LE, *NE, *NG, *NL, *CT, >, <, =, >=, <=, ¬=, ¬>, ¬<
7	*AND, &
8	*OR, *XOR, v, &&

Except for operators ¬ and *NOT, the operators for priorities 1 through 4 are numeric operators, which require numeric operands. The operators for priority 5 are string operators, which require operands to be either character or DBCS strings. Priority 6 operators are called relational operators, which require at least one operand that is a field name or a numeric or string expression (not a constant). The operators for priorities 7 and 8, plus the ¬ and *NOT operators (priority 1), are logical operators. The operands in a logical expression are relations (constructed by using a relational operator with appropriate operands) and other logical expressions.

The operands in a string expression, including string operands for a built-in function, are a combination of character fields and DBCS fields and constants. If both operands of such an expression are DBCS-only fields or constants, the final result from evaluation of the expression is a DBCS-only field value. If the operands are a combination of DBCS or character fields or constants, the result is a DBCS-open field value. When DBCS fields are concatenated, the extraneous shift-in and shift-out characters between the fields are removed.

The result produced by a + or - sign prefixed operator has the same attributes as the operand, unless the operand of a - sign prefixed operator is a *BIN2, in which case the result is a *BIN4. The result of an ** operator (exponentiation) is a double-precision floating-point number (*FLT8). For other numeric operators that require two operands, if either operand is a floating-point number, the result is a double-precision floating point number (*FLT8). If both operands are fixed-point numbers, the system uses the information in the following table to determine the number of total and fractional digits required to produce a packed decimal (*DEC) result. If both operands are zero-precision binary fields and/or integer constants, the result is a *BIN4, unless the operator is a "/". In that case, the result is the same as for a fixed-point result. If the total number of digits required exceeds 31, the number of fraction digits is reduced enough to enable calculation of the result with a total of 31 digits. If some fraction digits are dropped and the attributes of the end result of the computation (the attributes specified on the MAPFLD parameter for the field) require greater precision than that of the intermediate result, a warning message is sent to indicate that some precision was lost in evaluating the expression.

Operation	Result (Total Digits)	Result (Fractional Digits)
+	MAX(d1-f1,d2-f2)+MAX(f1,f2)+1	MAX(f1,f2)
-	MAX(d1-f1,d2-f2)+MAX(f1,f2)+1	MAX(f1,f2)
*	d1+d2	f1+f2
/	31	31-(d1-f1+f2)
//	MIN(d1-f1,d2-f2)+MAX(f1,f2)	MAX(f1,f2)

Legend:
d1 Total digits in operand 1
f1 Fractional digits in operand 1
d2 Total digits in operand 2
f2 Fractional digits in operand 2

When a numeric or string expression is specified on the MAPFLD parameter, the attributes of the final result are used in one of two ways. They are either used directly for the field value (if field-type *CALC is specified and the field is not contained in the prototype record format identified on the FORMAT parameter), or the final result is changed to match the attributes specified on the MAPFLD parameter or contained in the field definition in the record format identified by the FORMAT parameter.

Both operands of a relational operator can be constants. The fields, constants, or expressions specified as operands on the left and right side of a relational operator must be of the same type, either numeric or string. Any combination of character and DBCS field operands are allowed except that a character field cannot be related to a DBCS-only field.

There are two types of DBCS constants: DBCS-only and DBCS-open. A DBCS-only constant has only DBCS data between its apostrophes. This data must be enclosed in SO/SI characters. A DBCS-open constant has a mixture of DBCS and alphameric data. An SO character (HEX 0E) indicates the start of a group of DBCS characters and an SI character (HEX 0F) follows the last double-byte character of the group.

If a numeric or string expression appears as a complex selection operand on the QRYSLT or GRPSLT parameters, attributes of the final result of the expression used for the selection operand are changed to match the other relational operand.

It is not necessary for operands of a relational operator to have identical attributes, but numeric operands cannot be mixed with character operands. If the operands do not have identical attributes, the system changes them to identical attributes (except for the *CT operator, where the character string operands may be of different lengths), before performing the operation. This change uses packed decimal format if both operands are fixed-point numeric operands, or floating-point format if either operand is a floating-point number. The changes for fixed-point numeric operands align their decimal points and pad them with zeros. Numeric type changes may truncate fractional digits if more than 31 total digits are required for fixed-point numbers, or may drop some of the least significant digits if more than 15 total digits are required for floating-point numbers. Character operands are changed by padding the shorter operand with blanks.

The *CT operator performs a scan of the character field or string expression (except for expressions made up of a single character string literal) that must be specified as the left side of the relation, in order to determine if it contains the character string, field, or expression value specified as the right side of the relation. The second operand (the search value) must be no longer than the first operand (the base string).

If the string is found, the relation is satisfied and the result is a logical value of 'true'; otherwise, the result is a logical 'false' value. The following example illustrates this process:

Field BASEFLD contains the value 'THIS IS A TEST'.

Field TESTFLD contains the value 'TE'.

Expression	Result
'BASEFLD *CT "IS A"'	True
'BASEFLD *CT TESTFLD'	True
'BASEFLD *CT "X"'	False
'BASEFLD *CT TESTFLD vv "Z"'	False
'BASEFLD vv "ABC" *CT "TAB"'	True

Built-in Functions

The built-in functions listed below are supported for the expression used to define a derived field on the MAPFLD parameter or a complex selection operand specified on the QRYSLT or GRPSLT parameters.

A numeric argument is a numeric field, a numeric constant or a numeric expression. A string argument is a character field, a character string literal, or a string expression. Unless otherwise noted, all built-in functions allow expressions, including other built-in functions, to be used as arguments.

For a field that appears in the record format identified by the FORMAT parameter, and that is also defined by an expression on the MAPFLD parameter, the expression result is calculated by using the attributes described below. Then the resultant value is mapped to match the attributes of the field.

%ABSVAL (numeric-argument)

%ABSVAL accepts a numeric argument and returns the absolute value of the argument. The returned value has the same attributes as the argument, unless the argument is a *BIN2, in which case the returned value is a *BIN4.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a packed decimal number (*DEC) with 8 digits and 0 precision (date duration), 6 digits and 0 precision (time duration), or 20 digits and 6 precision (timestamp duration).

%ACOS (numeric-argument)

%ACOS accepts a numeric argument and returns the arc cosine of the argument, in radians. %ACOS and %COS are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%AND (string-argument ...)

%AND accepts two or more character or hexadecimal string arguments and returns a string that is the bit-wise 'AND' (logical and) of the arguments. This function takes the first argument string, ANDs it with the next string, and continues to AND

each successive argument with the previous result. If an argument is encountered that is shorter than the previous result, it is padded on the right with blanks. The returned value is a string of type *HEX with the same length as the longest argument. If any of the arguments are variable-length, the maximum length is used as the length of the argument.

%ANTILOG (numeric-argument)

%ANTILOG accepts a numeric argument and returns the antilogarithm (base 10) of the argument. %ANTILOG and %LOG are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%ASIN (numeric-argument)

%ASIN accepts a numeric argument and returns the arc sine of the argument, in radians. %ASIN and %SIN are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%ATAN (numeric-argument)

%ATAN accepts a numeric argument and returns the arc tangent of the argument, in radians. %ATAN and %TAN are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%ATANH (numeric-argument)

%ATANH accepts a numeric argument and returns the hyperbolic arc tangent of the argument, in radians. %ATANH and %TANH are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%AVG (numeric-argument)

%AVG accepts a numeric argument and returns the average value of its argument for the group of records defined on the GRPFLD parameter. The argument must be a field name or an expression (not a literal).

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. If no records are selected, the result is the null value. Otherwise,

- If the argument is fixed-point, the result is a packed decimal number (*DEC) with 31 total digits and the same number of integer digits as the argument.
- If the argument is floating-point, the result is a double-precision floating-point number (*FLT8).
- If the argument is date duration, time duration, or timestamp duration, the returned value is a packed decimal number (*DEC) with 31 digits and 0 precision (date duration), 31 digits and 0 precision (time duration), or 31 digits and 6 precision (timestamp duration).

%AVG is an aggregate function that is used for a nongrouping field in a query that uses the grouping function.

%CHAR (date/time-argument date/time-format)

%CHAR accepts a date/time argument and date/time format and returns the character representation of the argument using the specified format. The date/time argument can be a date, time, or timestamp field. The returned value is of type *CHAR and is tagged with the CCSID of the current job.

The date/time format is optional. If specified, it must be one of the following:

EUR European format
ISO International Standards Organization format
JIS Japanese Industrial Standard format
USA United States format

If the format is not specified, the job default format is used.

```
Example:
  OPNQRYF
     FILE(library/file)
     GRPFLD(charfld)
     GRPSLT('charfld = %CHAR(timefld "USA")')
```

%COS (numeric-argument)

%COS accepts a numeric argument and returns the cosine of the argument. The argument must be specified in radians. %COS and %ACOS are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%COSH (numeric-argument)

%COSH accepts a numeric argument and returns the hyperbolic cosine of the argument. The argument must be specified in radians.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%COT (numeric-argument)

%COT accepts a numeric argument and returns the cotangent of the argument. The argument must be specified in radians.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%COUNT

%COUNT does not support any arguments. It returns the count of the number of records contained in the group of records defined on the GRPFLD parameter. The returned value is a 4-byte binary number (*BIN4) with 10 total decimal digits and no fraction digits. %COUNT is an aggregate function that applies only to a query that uses the grouping function.

%CURDATE

%CURDATE does not support any arguments. It obtains the current date based on a reading of the time-of-day clock. The returned value is of type *DATE. The format and separator are derived from the job attributes.

%CURTIME

%CURTIME does not support any arguments. It obtains the current time based on a reading of the time-of-day clock. The returned value is of type *TIME. The format and separator are derived from the job attributes.

%CURTIMESTP

%CURTIMESTP does not support any arguments. It obtains the current timestamp based on a reading of the time-of-day clock. The returned value is of type *TIMESTP. The format and separator will be derived from the job attributes.

%CURTIMEZONE

%CURTIMEZONE does not support any arguments. It obtains the current time zone. The returned value is a packed decimal number (*DEC) with 6 digits and 0 precision.

%DATE (date/time-argument)

%DATE accepts a date/time argument and returns a date. The date/time argument can be a date or timestamp field, a character or hexadecimal field containing the external form of a date, a date literal, or a numeric field or literal value in the range 1 - 3,652,059. The returned value is of type *DATE.

```
Example:
  OPNQRYF
     FILE(library/file)
     QRYSLT(('%DATE(tstampfld) =
        "1989-10-23"'))
```

%DAY (date/time-argument)

%DAY accepts a date/time argument and returns the day part of the value. The date/time argument can be a date or timestamp field, a date duration or timestamp duration (field or literal), or a numeric field or literal. The returned value is of type *BIN4.

A numeric field argument must be defined as packed decimal (*DEC) with 8 digits and 0 precision for date duration or packed decimal (*DEC) with 20 digits and 6 precision for timestamp duration. A numeric constant argument must have 8 digits followed by a decimal point, or 14 digits followed by a decimal point and 6 digits.

%DAYS (date/time-argument)

%DAYS accepts a date/time argument and returns an integer representation of the date. The date/time argument can be a date or timestamp field, a character or hexadecimal field containing the external form of a date, or a date literal. The returned value is of type *BIN4.

%DIGITS (numeric-argument)

%DIGITS accepts a numeric argument and returns a character representation of its numeric value, not including the sign or a decimal point. The result is tagged with the CCSID of the current job. For example, %DIGITS (-1.5) returns the character string 15. The numeric argument must not be a floating point number.

%DURDAY (integer-argument)

%DURDAY accepts an integer argument and returns a labeled duration of days. The integer argument for this function can be a numeric expression, a field, or a literal.

This built-in function is allowed to stand by itself in the *mapped-field-definition* of the MAPFLD parameter, and is allowed as part of an arithmetic (addition or subtraction) expression with a date or timestamp field on the QRYSLT, GRPSLT, or MAPFLD parameters.

%DURHOUR (integer-argument)

%DURHOUR accepts an integer argument and returns a labeled duration of hours. The integer argument for this function can be a numeric expression, a field, or a literal.

This built-in function is allowed to stand by itself in the *mapped-field-definition* on the MAPFLD parameter, and is allowed as part of an arithmetic (addition or subtraction) expression with a time or timestamp field on the QRYSLT, GRPSLT, or MAPFLD parameters.

%DURMICSEC (integer-argument)

%DURMICSEC accepts an integer argument and returns a labeled duration of microseconds. The integer argument for this function can be a numeric expression, a field, or a literal.

This built-in function is allowed to stand by itself in the *mapped-field-definition* on the MAPFLD parameter, and is allowed as part of an arithmetic (addition or subtraction) expression with a timestamp field on the QRYSLT, GRPSLT, or MAPFLD parameters.

%DURMINUTE (integer-argument)

%DURMINUTE accepts an integer argument and returns a labeled duration of minutes. The integer argument for this function can be a numeric expression, a field, or a literal.

This built-in function is allowed to stand by itself in the *mapped-field-definition* on the MAPFLD parameter, and is allowed as part of an arithmetic (addition or subtraction) expression with a time or timestamp field on the QRYSLT, GRPSLT, or MAPFLD parameters.

%DURMONTH (integer-argument)

%DURMONTH accepts an integer argument and returns a labeled duration of months. The integer argument for this function can be a numeric expression, a field, or a literal.

This built-in function is allowed to stand by itself in the *mapped-field-definition* on the MAPFLD parameter, and is allowed as part of an arithmetic (addition or subtraction) expression with a date or timestamp field on the QRYSLT, GRPSLT, or MAPFLD parameters.

%DURSEC (integer-argument)

%DURSEC accepts an integer argument and returns a labeled duration of seconds. The integer argument for this function can be a numeric expression, a field, or a literal.

This built-in function is allowed to stand by itself in the *mapped-field-definition* on the MAPFLD parameter, and is allowed as part of an arithmetic (addition or subtraction) expression with a time or timestamp field on the QRYSLT, GRPSLT, or MAPFLD parameters.

%DURYEAR (integer-argument)

%DURYEAR accepts an integer argument and returns a labeled duration of years. The integer argument for this function can be a numeric expression, a field, or a literal.

This built-in function is allowed to stand by itself in the *mapped-field-definition* value on the MAPFLD parameter, and is allowed as part of an arithmetic (addition or subtraction) expression with a date or timestamp field on the QRYSLT, GRPSLT, or MAPFLD parameters.

```
Example:
 OPNQRYF
   FILE((library/file))
   QRYSLT('startfld > %CURDATE + oneyear *AND
           endfld   < %CURDATE + %DURYEAR(2)')
   MAPFLD((oneyear '%DURYEAR(1)'))
```

%EXP (numeric-argument)

%EXP accepts a numeric argument and returns a value that is the base of the natural logarithm (e) raised to a power specified by the argument. %EXP and %LN are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types may be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%HEX (hexadecimal-argument)

%HEX accepts an argument and returns the hexadecimal equivalent of the argument's value. The hexadecimal argument can be of any type. The returned value is of type *CHAR, and is tagged with the CCSID of the current job.

%HOUR (date/time-argument)

%HOUR accepts a date/time argument and returns the hour part of the value. The date/time argument can be a time or timestamp field, a time duration or timestamp duration (either field or literal), or a numeric field or literal. The returned value is of type *BIN4.

A numeric field argument must be defined as packed decimal (*DEC) with 6 digits and 0 precision for time duration or packed decimal (*DEC) with 20 digits and 6 precision for timestamp duration. A numeric constant argument must have 6 digits followed by a decimal point, or 14 digits followed by a decimal point and 6 digits.

```
Example:
  OPNQRYF
     FILE(library/file)
     QRYSLT(('%HOUR(timefld2) = 12'))
```

%LEN (length-argument)

%LEN accepts one argument and returns the number of bytes used to represent the value unless the value is a graphic field type. If the value is a graphic field type, the number of graphic characters is returned. The length argument can be of any type. The returned value is of type *BIN4.

```
Example:
  OPNQRYF
     FILE(library/file)
     QRYSLT('%LEN(varlenfld) <= 30')
```

Argument Type	Result Length in Bytes
Character	1-32766
Hex	1-32766
DBCS-only	4-32766
DBCS-either	4-32766
DBCS-open	4-32766
DBCS-graphic	1-16383
Variable Character	0-32740
Variable Hex	0-32740
Variable DBCS-only	0-32740
Variable DBCS-either	0-32740
Variable DBCS-open	0-32740
Variable DBCS-graphic	0-16370
Date	4
Time	3
Timestamp	10
Binary *BIN4	2
Binary *BIN8	4
Floating point *FLT4	4
Floating point *FLT8	8
Packed decimal (p,s)	INTEGER(p/2)+1, (1-16)
Zoned decimal (p,s)	p (1-31)

p=precision, s=scale

String notes:

The %LEN function returns the length of the value as it is stored in the data space.

- For fixed-length fields, the length is always the same as the declared size of the field, not the length of the actual data in the field.
- For variable-length fields, the length is the length of the actual data in the field, including trailing blanks.

For example, assume FIXED10 is a *CHAR(10) field, and VAR10 is a *VCHAR(10) field. The following example shows results of the %LEN function:

%LEN Statement	Field Data	Result
%LEN(fixed10)	'1234567890'	10
%LEN(fixed10)	'12345'	10
%LEN(var10)	'1234567890'	10
%LEN(var10)	'12345'	5
%LEN(var10)	'12345 '	7
%LEN(var10)	''	0

%LN (numeric-argument)

%LN accepts a numeric argument and returns the natural logarithm of the argument. %LN and %EXP are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types may be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%LOG (numeric-argument)

%LOG accepts a numeric argument and returns the common logarithm (base 10) of the argument. %LOG and %ANTILOG are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types may be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%MAX (numeric-or-string-or-date/time-argument ...)

%MAX accepts one or more character-string, DBCS-string, numeric, or date/time arguments, and returns the largest value from the list. Date/time arguments are arguments of type *DATE, *TIME, or *TIMESTP, or arguments that are date, time, or timestamp durations. String arguments must be no longer than 256 bytes.

If only one argument is specified, this function returns the maximum value of its argument for the group of records defined on the GRPFLD parameter, and the returned value has the same attributes as the argument. If no records are selected, the result is the null value. If the single argument is a date duration, time duration, or timestamp duration, then the returned value is a packed decimal number (*DEC) with 8 digits and 0 precision (date duration), 6 digits and 0 precision (time duration), or 20 digits and 6 precision (timestamp duration). When a single argument is used, it must be a field name or an expression (not a literal). %MAX with only one argument is an aggregate function that is used for a nongrouping field in a query that uses the grouping function.

If multiple arguments are specified, %MAX returns the maximum value of all the arguments. All arguments must be either character-string, DBCS-string, numeric, or date/time values. This function calculates the maximum value of the first two arguments, and then continues to determine the maximum value of the previous result and the next successive argument. The final result is determined according to the following value conversion rules.

If an argument has different attributes than the previous result, the two values are converted to identical attributes and the operation continues. This conversion uses packed decimal if both values are fixed-point numeric values, or floating-point if either value is floating-point. The conversion for fixed-point numeric values aligns the decimal points and pads the values with zeros. Numeric type changes may truncate fractional digits if more than 31 total digits are required for fixed-point numbers, or drop some of the least significant digits if more than 15 total digits are required for floating-point numbers. Character values are changed by padding the shorter field with blanks.

%MICSEC (date/time-argument)

%MICSEC accepts a date/time argument and returns the microsecond part of the value. The date/time argument can be a timestamp (field or literal), a timestamp duration (field or literal), a character field that contains the external form of a timestamp, or a numeric field or literal. The returned value is of type *BIN4.

A numeric field argument must be defined as packed decimal (*DEC) with 20 digits and 6 precision for timestamp duration. A numeric constant argument must be 14 digits followed by a decimal point and 6 digits.

%MIN (numeric-or-string-or-date/time-argument ...)

%MIN accepts one or more character-string, DBCS-string, numeric, or date/time arguments, and returns the smallest value from the list. Date/time arguments are arguments of type *DATE, *TIME, or *TIMESTP, or arguments that are date, time, or timestamp durations. String arguments must be no longer than 256 bytes.

If only one argument is specified, this function returns the minimum value of its argument for the group of records defined on the GRPFLD parameter, and the returned value has the same attributes as the argument. If no records are selected, the result is the null value. If the single argument is a date duration, time duration, or timestamp duration, then the returned value is a packed decimal number (*DEC) with 8 digits and 0 precision (date duration), 6 digits and 0 precision (time duration), or 20 digits and 6 precision (timestamp duration). When a single argument is used, it must be a field name or an expression (not a literal). %MIN with only one argument is an aggregate function that is used for a nongrouping field in a query that uses the grouping function.

If multiple arguments are specified, %MIN returns the minimum value of all the arguments. All arguments must be either character-string, DBCS-string, numeric, or date/time values. This function calculates the minimum value of the first two arguments, and then continues to determine the minimum value of the previous result and the next successive argument. The final result is determined by the value change rules described below.

If an argument has different attributes than the previous one, the two values are changed to identical attributes and the operation continues. This change uses packed decimal numbers if both values are fixed-point numeric values, or floating-point numbers if either value is a floating-point number. The change for fixed-point numeric values aligns the decimal points and pads with zeros. Numeric type change may truncate fractional digits if more than 31 total digits are required for fixed-point numbers, or may drop some of the least significant digits if more than 15 total digits are required for floating-point numbers. Character values are changed by padding the shorter field with blanks.

%MINUTE (date/time-argument)

%MINUTE accepts a date/time argument and returns the minute part of the value. The date/time argument can be a time or timestamp field, a time duration or timestamp duration (either field or literal), or a numeric field or literal. The returned value is of type *BIN4.

A numeric field argument must be defined as packed decimal (*DEC) with 6 digits and 0 precision for time duration or packed decimal (*DEC) with 20 digits and 6 precision for timestamp duration. A numeric constant argument must have 6 digits followed by a decimal point, or 14 digits followed by a decimal point and 6 digits.

%MONTH (date/time-argument)

%MONTH accepts a date/time argument and returns the month part of the value. The date/time argument can be a date or timestamp field, a date duration or timestamp duration (field or literal), or a numeric field or literal. The returned value is of type *BIN4.

A numeric field argument must be defined as packed decimal (*DEC) with 8 digits and 0 precision for date duration or packed decimal (*DEC) with 20 digits and 6 precision for timestamp duration. A numeric constant argument must have 8 digits followed by a decimal point, or 14 digits followed by a decimal point and 6 digits.

%NONNULL (argument ...)

%NONNULL accepts a list of two or more arguments and returns the first non-null value from the list. The items in the argument list can be fields or literal values of any type. The type of the returned value is that of the item selected from the list.

```
Example:
  OPNQRYF
     FILE(library/file)
     QRYSLT('%NONNULL(fld1 fld2 0) > 0')
```

The above example selects records from the file where either field FLD1 or field FLD2 contains a non-null value that is greater than zero. If both FLD1 and FLD2 were null, the %NONNULL function specified in this example would return '0' because of the constant '0' passed as the third argument. If any field is DBCS-graphic, all fields must be DBCS-graphic.

%NOT (string-argument)

%NOT accepts a character or hexadecimal string argument and returns a string that is the bit-wise 'NOT' (logical not) of the argument. The returned value is a string of type *HEX with the same length as the argument.

%OR (string-argument ...)

%OR accepts two or more character-string arguments and returns a string that is the bit-wise 'OR' (logical inclusive or) of the arguments. This function takes the first argument string, ORs it with the next string, and then continues to OR each successive argument with the previous result. If an argument is encountered that is shorter than the previous result, it is padded with blanks. The final result is a string with the same length as the longest argument. If any of the arguments are variable-length, the maximum length is used as the length of the argument.

%SECOND (date/time-argument)

%SECOND accepts a date/time argument and returns the seconds part of the value. The date/time argument can be a time or timestamp field, a time duration or timestamp duration (either field or literal), or a numeric field or literal. The returned value is of type *BIN4.

A numeric field argument must be defined as packed decimal (*DEC) with 6 digits and 0 precision for time duration or packed decimal (*DEC) with 20 digits and 6 precision for timestamp duration. A numeric constant argument must have 6 digits followed by a decimal point, or 14 digits followed by a decimal point and 6 digits.

%SIN (numeric-argument)

%SIN accepts a numeric argument and returns the sine of the argument. The argument must be specified in radians. %SIN and %ASIN are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%SINH (numeric-argument)

%SINH accepts a numeric argument and returns the hyperbolic sine of the argument. The argument must be specified in radians.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%SQRT (numeric-argument)

%SQRT accepts a numeric argument and returns the square root of the argument.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%SST (string-argument start-position-expression <length-expression>)

%SST and %SUBSTRING accept a character, hexadecimal, DBCS, or graphic string, a starting position expression, and an optional length expression as arguments. They return a substring of the string argument that is of the same type and CCSID as the string argument and has length equal to the value specified by the length-expression.

Single-byte substringing is done when these functions (%SST and %SUBSTRING) are used for DBCS data. The shift-out and shift-in characters may be lost, which produces unusual results. The result of the DBCS substring operation is the DBCS-open type.

The string argument can be a fixed- or variable-length character, hexadecimal, DBCS, or graphic field or an expression which evaluates to a fixed- or variable-length character, hexadecimal, DBCS, or graphic string.

The values derived from expressions for the second and third arguments must be valid integers. The second argument must have a value between 1 and the length attribute (or maximum length of a variable-length field) of the first argument, and the third argument must have a value between 1 and the length attribute (or maximum length of a variable-length field) of the first argument.

If an argument is DBCS-graphic, the second and third arguments must also be specified as DBCS-graphic characters, not bytes.

If an expression is given for the second or third arguments, the expression must be enclosed in parentheses.

If the expressions evaluate to variable-length results, no validation of the range of these expressions is guaranteed and errors may occur during input/output processing.

The maximum value allowed for the third argument (length) is 32766 except for DBCS-graphic, which is 16383. However, if the third operand is represented by an expression, this causes the result to be variable-length. Thus, the value of the expression cannot exceed 32740 except for DBCS-graphic, which cannot exceed 16370.

The user can omit the third argument. If the third argument is not specified and the first argument is:

- fixed-length, the default value for the third argument is LENGTH(argument_1) - value_for_argument_2 + 1
- variable-length, the default value for the third argument is the maximum of 0 and

LENGTH(argument_1) - value_for_argument_2 + 1

- variable-length with a length less than the value for argument_2, the default value for the third argument is zero and the result is the empty string.

```
Example:
  OPNQRYF
     FILE(library/file)
     QRYSLT('field1 =
     %SST(field2 (numfld1+3)
     (numfld1+numfld2))')
```

%STDDEV (numeric-argument)

%STDDEV accepts a numeric argument and returns the standard deviation of its argument for the group of records defined by the GRPFLD parameter. The argument must be a field name or an expression (not a literal). If no records are selected, the result is the null value. Otherwise, the returned value is a double-precision floating-point number (*FLT8). %STDDEV is an aggregate function that is used for a nongrouping field in a query that uses the grouping function.

%STRIP(string-argument <strip-character> <strip-function>)

%STRIP accepts a character-, DBCS-, or graphic-string argument, an optional strip character, and an optional strip function as arguments. It returns a result string with the strip character removed from the string argument as specified by the strip function.

The string argument can be a literal, a fixed or variable-length character, hexadecimal, DBCS, or graphic field, or an expression which evaluates to a fixed- or variable-length character, hexadecimal, DBCS, or graphic string.

The strip character must be a single character, enclosed in apostrophes, with a data type compatible to the source string. The default is a single SBCS space for character data, DBCS-open, and DBCS-either, a single DBCS space for DBCS-only data, and a single graphic space for graphic data.

The strip function can be one of three functions:

***LEAD** Remove leading strip character(s)
***TRAIL** Remove trailing strip character(s)
***BOTH** Remove both leading and trailing strip character(s)

The default strip function is *BOTH.

The return value is a variable-length string with the same type, CCSID, and maximum length as the string argument. If the source string or strip character is null, the result is null.

```
Example:
  OPNQRYF
     FILE(library/file)
     QRYSLT('%STRIP(fld '.' *TRAIL) = 'Mr')
```

%SUBSTRING (string-field-name start-position length)

%SUBSTRING performs the same operation as %SST. See the %SST description for more information.

%SUM (numeric-argument)

%SUM accepts a numeric argument and returns the sum of all the values for its argument in the group of records defined on the GRPFLD parameter. The argument must be a field name or an expression (not a literal).

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. If no records are selected, the result is the null value. Otherwise,

- If the argument is floating-point number, the returned value is a double-precision floating-point number (*FLT8).
- If the argument is a binary number with zero-precision, the returned value is *BIN4.
- If the argument is a binary number with nonzero precision or a fixed-point number, the returned value is a packed decimal number (*DEC) with 31 total digits and as many fractional digits as the argument.
- If the argument is of type date duration, time duration, or timestamp duration, the returned value is a double-precision floating-point number (*FLT8).

%SUM is an aggregate function that is used for a nongrouping field in a query that uses the grouping function.

%TAN (numeric-argument)

%TAN accepts a numeric argument and returns the tangent of the argument. The argument must be specified in radians. %TAN and %ATAN are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The return value is a double-precision floating-point number (*FLT8).

%TANH (numeric-argument)

%TAN accepts a numeric argument and returns the hyperbolic tangent of the argument. The argument must be specified in radians. %TANH and %ATANH are inverse operations.

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. Arguments of these types can be specified either as fields or literal values. The returned value is a double-precision floating-point number (*FLT8).

%TIME (date/time-argument)

%TIME accepts a date/time argument and returns a time. The date/time argument can be a time or timestamp field, a character or hexadecimal field containing the external form of a time, or a time literal. The returned value is of type *TIME.

%TIMESTP (date/time-argument date/time-argument)

%TIMESTP accepts one or two date/time arguments and returns a timestamp.

- If only one date/time argument is specified, it must be a timestamp (field or literal), or a character or hexadecimal field containing the external form of a timestamp.
- If both arguments are specified,
  1. The first date/time argument must be a date (field or literal), or a character or hexadecimal field containing the external form of a date.
  2. The second date/time argument must be a time (field or literal), or a character or hexadecimal field containing the external form of a time.

The returned value is of type *TIMESTP.

%USER

%USER does not support any arguments. It returns the user profile name of the job in which the query is running. The returned value is of type variable-length character (*VCHAR) with a maximum length of 18.

```
Example:
  OPNQRYF
     FILE(library/file)
     QRYSLT('field = %USER')
```

%VAR (numeric-argument)

%VAR accepts a numeric argument and returns the variance of its argument for the group of records defined by the GRPFLD parameter. The argument must be a field name or an expression (not a literal).

The following argument types are treated as numeric values: date duration, time duration, and timestamp duration. If no records are selected, the result is the null value. Otherwise, the returned value is a double-precision floating-point number (*FLT8). %VAR is an aggregate function that is used for a nongrouping field in a query that uses the grouping function.

%XLATE (string-argument qualified-table)

%XLATE accepts a character-string argument and the name of a table object (*TBL), and returns a string that is the value of the first argument translated by using the contents of the table. The returned value is a string with the same length and CCSID as the first argument.

The second argument must be a simple or qualified table object name. If no library name is specified, *LIBL is used to find the table.

%XOR (string-argument...)

%XOR accepts two or more character-string arguments and returns a string that is the bit-wise 'XOR' (logical exclusive or) of the arguments. This function takes the first argument string, XORs it with the next string, and then continues to XOR each successive argument with the previous result. If an argument is encountered that is longer than the previous result, the previous result is padded with blanks before the XOR operation. If any of the arguments is variable-length, the maximum length is used as the length of the argument. The final result is a string of type *HEX with the same length as the longest argument.

%YEAR

%YEAR accepts a date/time argument and returns the year part of the value. The date/time argument can be a date or timestamp field, a date duration or timestamp duration (field or literal), or a numeric field or literal. The returned value is of type *BIN4.

A numeric field argument must be defined as packed decimal (*DEC) with 8 digits and 0 precision for date duration or packed decimal (*DEC) with 20 digits and 6 precision for timestamp duration. A numeric constant argument must have 8 digits followed by a decimal point, or 14 digits followed by a decimal point and 6 digits.

Restricted Built-in Functions

The following built-in function is supported only as the second operand of the 'equal' or 'not-equal' relational operators specified on the QRYSLT or GRPSLT parameter.

%NULL

%NULL accepts no arguments. It is used to select or omit records based on whether or not a field in the record contains a null value.

```
Example:
  OPNQRYF
     FILE(library/file)
     QRYSLT('charfld = %NULL')
```

This query would select all the records where 'charfld' contains the null value.

The following three built-in functions are supported only as the second operand of the 'equal' relational operator specified on the QRYSLT or GRPSLT parameter.

%RANGE (low-value high-value)

%RANGE is used to identify the lower and upper boundaries for the value of a field or expression. %RANGE must be specified as the right side of a relation whose operator is equal. The low-value and high-value argument must be field names, character strings, or numeric literals, to match the type of field or expression specified as left side of the relation.

For example, to select only records where the numeric field NBRFLD has a value ranging from 10 through 20, specify:

```
'nbrfld = %RANGE(10 20)'
```

If the low-value argument is greater than the high-value argument, the relation produces a logical value of 'false'.

%VALUES (allowed-value...)

%VALUES is used to identify a list of allowed values for a field or expression. %VALUES must be specified as the right side of a relation whose operator is equal. The allowed-value arguments must be character string or numeric literals, to match the type of the field or expression specified as the left side of the relation. For example, to select only records where the second character of field CHARFLD has a value that is one of the values 'A', 'E', 'I', 'O', or 'U', specify the following:

```
'%SST(charfld 2 1) =
  %VALUES(''A'' ''E'' ''I'' ''O'' ''U'')'
```

%WLDCRD ("pattern-string" "wild-characters"])

%WLDCRD is used to specify a pattern that performs a wildcard scan of the character or hexadecimal field or string expression (except for expressions made up of a single character-string literal) that must be specified as the left side of the relation. %WLDCRD must be specified as the right side of a relation whose operator is equal. The pattern-string argument must be a character-string, DBCS, or graphic literal, to match the left side of the relation. The wild-characters argument is an optional parameter that specifies what 'wildcard' characters are used in the pattern-string.

If specified for character data only (no DBCS data), the wild-characters argument must be a character-string literal of exactly two characters. The first character is the value that matches any single character in the search string. The second character is the value that matches a substring of any zero or more characters. The two characters must not be the same, but there is no requirement that either character appear in the pattern-string. If the wild-characters argument is omitted, the default is for an underline ('_') to match any single character and an asterisk ('*') to match a substring of any zero or more characters.

If the wild-characters argument is specified for DBCS data only (no character data), the argument must be a double-byte character-string literal of exactly two double-byte characters. The first double-byte character is the value that will match any one double-byte character in the search string. The second double-byte character is the value that will match a substring of any zero or more characters. The two double-byte characters must not be the same, but there is no requirement that either character appear in the pattern string. If the wild-characters argument is omitted, the default is for a DBCS underline to match any one double-byte character and a DBCS asterisk to match a substring of any zero or more double-byte characters.

If the wild-characters argument is specified for both character and DBCS data, in addition to the previous rules, the argument must first contain a single-byte character-string literal (two single-byte characters), then a double-byte character string (two double-byte characters).

In this case, the first character matches any single-byte character in the character string, the second character matches a substring of any number of single-byte or double-byte characters. The first double-byte character matches any double-byte character in the character string. The second double-byte character matches a substring of any number of single-byte or double-byte characters.

The following example selects only records where the character field CHARFLD contains a 'T', followed by any two characters and an 'E', appearing anywhere in the field.

```
'charfld = %WLDCRD(''*T__E*'')'
```

Note: The asterisks at the start and end of the pattern-string are required to allow the 'T' and 'E' to appear somewhere other than the first and last positions in the field:

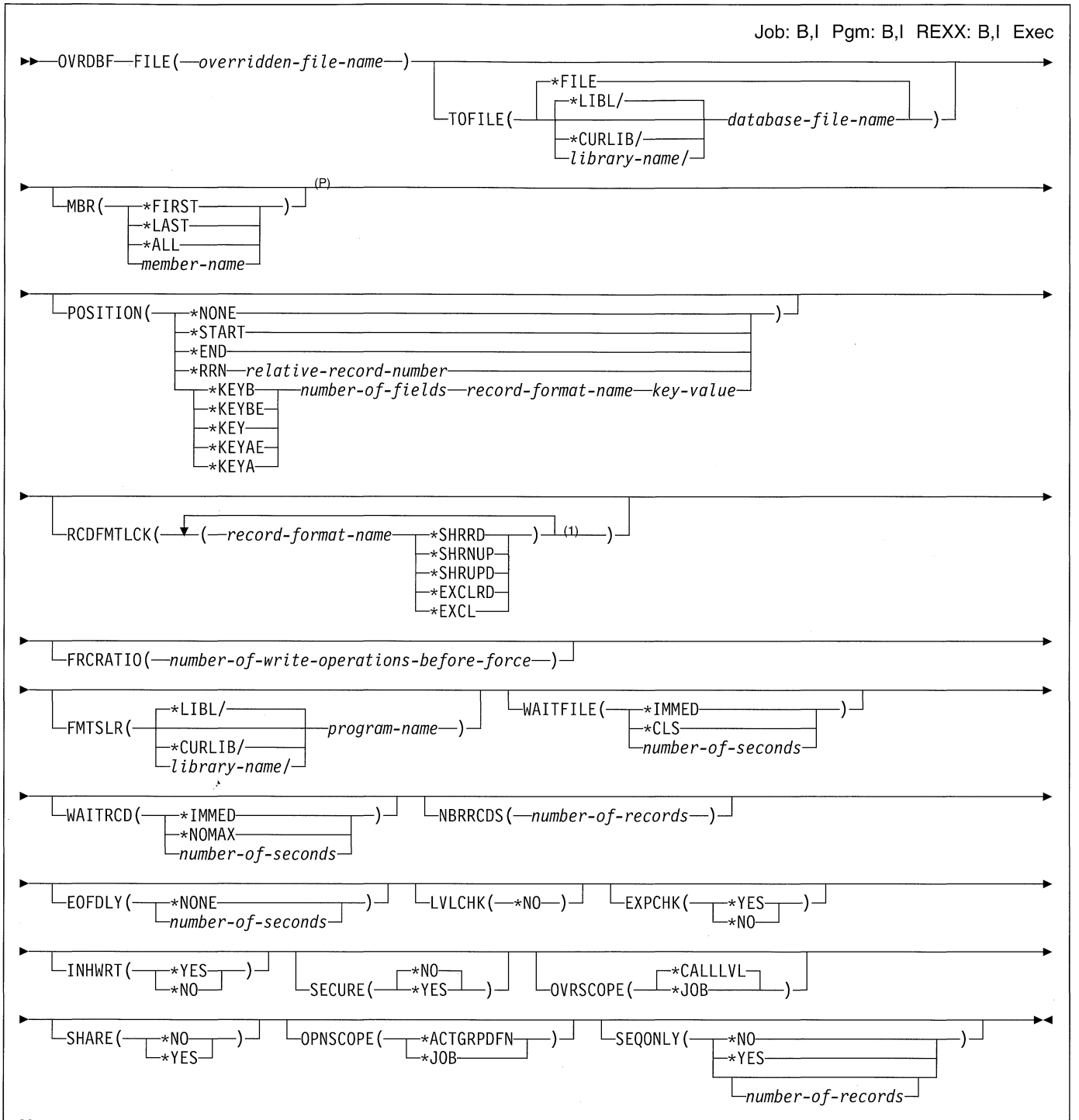
To select only records where the character field CHARFLD starts with the string 'ABC', followed by one or more other characters and then followed by the string 'XYZ' (but not necessarily at the end of the field), specify the following:

```
'charfld = %WLDCRD(''ABC_*XYZ*'')'
```

To select only records where the second character of field CHARFLD is an asterisk ('*'), the last character is an underline ('_'), and the letter 'M' appears somewhere in between, specify the following:

```
'charfld = %WLDCRD(''#*.M._'' ''#.'')'
```

OVRDBF (Override with Database File) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 A maximum of 32 repetitions

Purpose

The Override with Database File (OVRDBF) command is used to:

- Override (replace) the file named in the program
- Override certain parameters of a file that are used by the program
- Override the file named in the program and override certain parameters of the file processed

Parameters overridden by this command are specified in the file description, in the program, and/or in other previously issued file override commands. The OVRDBF command applies to physical files, logical files, and distributed data management (DDM) files.

To override (replace) a file named in the program, specify the name of that file in the FILE parameter, and specify the name of the file that overrides it (the file to be processed by the program) in the TOFILE parameter. The other parameters of this command can be used to override parameter values contained in the file description of the overriding file.

To override only certain parameters of the file named in the program, instead of replacing the entire file, specify the name of the file in the FILE parameter and specify the *FILE value for the TOFILE parameter. Then use the other parameters of this command to override specific parameters of the file. Parameters that are not specified do not affect parameters specified in the file description, in the program, or in other previously issued file override commands.

Note: The override cannot be used for all commands. A list of the commands that cannot be overridden, along with more information on overriding files is in the *Data Management Guide*.

Required Parameter

FILE

Specifies the name of the file being used by the program to which this override command is applied. If TOFILE(*FILE) is specified, a display device file must be specified. Otherwise, any device file or database file can be specified.

Optional Parameters

TOFILE

Specifies the qualified name of the database file that is used instead of the file specified in the FILE parameter or, if *FILE is specified, specifies that certain attributes are overridden by parameters specified in this command. The parameters specified on this OVRDBF command override the same parameters specified in the database file, in the program, and/or in other previously issued OVRDBF commands.

***FILE:** The database file named in the FILE parameter has some of its parameters overridden by values specified in this command.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file that is used instead of the file specified in the FILE parameter.

MBR

Specifies the members used within the database file. This parameter is not valid for DDM files that reference remote systems other than the System/38 or the AS/400 system.

***FIRST:** The first member in the database file is used.

***LAST:** The last member of the specified physical file is used.

***ALL:** All members in the file are processed sequentially. All members are opened with the same override parameters as the first member. While overrides issued prior to the open operation of the first member are processed, overrides or delete overrides issued following the open operation of the first member are not processed. EOFDLY, FMTSLR, INHWRT, or the POSITION parameter cannot be specified if MBR(*ALL) has been specified on a previously issued OVRDBF command that is still in effect for this file. An escape message is sent if any of the mutually exclusive parameters are specified.

member-name: Specify the member name that overrides (at file open time) the member name specified in the using program, and/or in other called OVRDBF commands. If the member name is not specified, and a TOFILE parameter other than *FILE has been specified, the first member in the file is used.

POSITION

Specifies the starting position for retrieving records from the database file. The first record to get can be at the beginning (*START) or at the end (*END) of the file, the nth record in the file (*RRN), or the record indicated by a key field value and one of the key-search values (*KEY, *KEYA, *KEYAE, *KEYB, or *KEYBE). This parameter overrides the value specified in the program, and/or in other called OVRDBF commands.

Note: This parameter cannot be specified if MBR(*ALL) was specified in a previously issued OVRDBF command that is still in effect for this file.

***NONE:** No special positioning is required. The first I/O operation indicates the record that is retrieved.

***START:** The starting position is the first record in the file. If a read-previous is specified in the program, an end-of-file condition occurs.

***END:** The starting position is the last record in the file. When the next record is retrieved, an end-of-file condition is reached. If a read previous is requested, the last record of the file is retrieved.

Element 1: Relative Record Number

***RRN** *relative-record-number:* Specify the number of the relative record (its position from the beginning of the file), preceded by the value *RRN, that is retrieved first. For example, POSITION(*RRN 480) specifies that record number 480 is retrieved next. If a read-previous is requested, the 479th record in the file is retrieved.

Element 2: Key-Search Values

The first record that is retrieved is identified by the specified key-operation, number-of-fields, record-format-name, and key-value. If a record that matches these values does not exist, an error message is sent.

Specify one of the following key-search types:

***KEYB** *(key-before):* A record that precedes the record identified by the remaining search values (number-of-fields, record-format-name, and key-value) is the first record retrieved.

***KEYBE** *(key-before or equal):* The record identified by the search values is the first record retrieved. If no record matches those values, the record is selected that matches the largest previous value.

***KEY** *(key-equal):* The record identified by the search values is the first record retrieved. If a read-previous is specified in the program, the preceding record is retrieved.

***KEYAE** *(key-after or equal):* The record identified by the search values is the first record retrieved. If no records matches those values, the record is selected with the next highest value.

***KEYA** *(key-after):* A record that follows the record identified by the remaining search values (number-of-fields, record-format-name, and key-value) is the first record retrieved.

Specify the remaining search values as follows:

Element 3: Number of Fields

number-of-fields: Specify the number of key fields to use in the search. The number of fields specified in this parameter does not have to be the same as the actual number of fields in each key for the file. For example, if POSITION(*KEY 1 FMT1 A) is specified, the first record in the file format FMT1 that has a first key field value of A is retrieved. If a number of fields of zero are specified, the search is based on all key fields. If zero is used, the key value contains the maximum key size.

Element 4: Name of Record Format

record-format-name: Specify the name of the record format in the database file that contains the key value specified. If no record format name is specified (*N), all record formats are searched for the first record that matches the other search values.

Element 5: Key Value

key-value: Specify the first record retrieved. This value is specified as a quoted character string for character or positive zoned decimal formats, or is specified in hexadecimal form at (x'value'). You can specify up to 2000 characters in the character string.

For example, POSITION(*KEY 1 FMT2 X'123F') specifies that the system searches for a record from the record format FMT2, that a single key field is used in the search (even though the key value may have more key fields), and that the record contains the hexadecimal value 123F (the hexadecimal equivalent of packed decimal value 123).

POSITION(*KEYB 0 *N X'123F') specifies that a record of any format is retrieved next (its key value must precede the record identified by key value X'123F').

If a key is specified that contains more than one field, the key must be coded to match the definition of the key in the file. If the definition is for a key other than a character or signed decimal key, the key must be coded in hexadecimal form.

For example, suppose the key definition has the following key fields:

- Character field (6A)
- Packed numeric field (5P 2)
- Signed numeric field (2S 0)

A character string the length of the entire key (6+3+2 in this example) can be specified on the POSITION parameter. POSITION(*KEY 3 YOURFMT X'E6D9C5D5C3C812345FF9F9') specifies that the system searches for the record in format YOURFMT, a key containing three fields is used in the search, and the record contains the hexadecimal value E6D9C5D5C3C812345FF9F9. The hexadecimal value corresponds to the following desired key values:

- Hexadecimal value E6D9C5D5C3C8 corresponds to the character field key value WRENCH.
- Hexadecimal value 12345F corresponds to the packed numeric field value +123.45.
- Hexadecimal value F9F9 corresponds to the signed numeric field value 99.

The *DDM Guide* has more information on the effects of using the POSITION parameter with DDM files.

RCDFMTLCK

Specifies the lock state of the named record format while it is used by the program. The lock state indicates how the data associated with each format is locked. The following chart shows the lock states that are specified for

each record format and the operations allowed to other programs when the lock is in effect:

Lock State	Definition	Other Program Operations
*SHRRD	Shared read	Read and update allowed
*SHRNUP	Shared read, no update	Read allowed, update not allowed
*SHRUPD	Shared update	Read and update allowed
*EXCLRD	Exclusive allow read	Read allowed, update not allowed
*EXCL	Exclusive no read	Neither read nor update allowed

An explanation of each lock state is in the *CL Programmer's Guide*.

For each record format, specify the record format name followed by one lock state value. This parameter overrides the record format locks specified in the program, in other called OVRDBF commands, and the default locks established when the member was created. If the lock state specified for the file in an Allocate Object (ALCOBJ) command is more restrictive than the lock state specified in this parameter, this parameter is ignored. Therefore, this parameter can only impose a more restrictive lock state on a record format than that specified for the file.

FRCRATIO

Specifies the number of inserts, deletes, or updates that occur to records before they are forced into auxiliary (permanent) storage. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

If a physical file associated with this database file is journaled, specify a larger force-write ratio. More information on journal management is in the *Advanced Backup and Recovery Guide*.

This parameter overrides the force-write ratio specified in the database file, in the program, and/or in other previously issued OVRDBF commands.

FMTSLR

Specifies the qualified name of a record format selection program that is called when a logical file member contains more than one logical record format. The user-written selector program is called when a record is inserted into the database file and a record format name is not specified in the high-level language program. More information about the use of format selector programs is in the *Database Guide*. This parameter overrides the value specified in the database file and in other previously issued OVRDBF commands.

A program specified as the format selector program cannot be created with USRPRF(*OWNER) specified in the Create CL Program (CRTCLPGM) command.

Note: This parameter cannot be specified if MBR(*ALL) was specified in a previously issued OVRDBF command that is still in effect for this file.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of a record format selection program called when a logical file member contains more than one logical record format.

WAITFILE

Specifies the number of seconds that the program waits for the file resources and session resources to be allocated when the file is opened, or for the device or session resources to be allocated when an acquire operation is performed to the file. If those resources are not allocated within the specified wait time, an error message is sent to the program. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: An immediate allocation of the device by the device resource is required when an acquire operation is performed to the file.

This parameter overrides the wait time specified in the database file, in the program, and/or in other previously issued OVRDBF commands.

***IMMED:** The program does not wait; when the file is opened, an immediate allocation of the file resources is required.

***CLS:** The job default wait time is used as the wait time for the file resources being allocated.

number-of-seconds: Specify the number of seconds that the program waits for the file resources to be allocated. Valid values range from 1 through 32767 seconds.

WAITRCD

Specifies the number of seconds that a program waits for a record to be updated or deleted, or for a record read in the commitment control environment with LCKLVL(*ALL) specified. More information on record locking is in the *Database Guide*. If the record is not allocated in the specified wait time, an error message is sent to the program.

Note: This parameter overrides the record wait time specified in the database file, specified in the program, and/or in other previously issued OVRDBF commands. The minimum delay for DDM files is 60 seconds. This value may need to be longer than the delay specified for local database files.

***IMMED:** The program does not wait; when a record is locked, an immediate allocation of the record is required.

***NOMAX:** There is no disconnect limit.

number-of-seconds: Specify the number of seconds that the program waits for the record lock. Valid values range from 1 through 32767 seconds.

NBRRCDS

Specifies the number of records moved as a unit from auxiliary storage to main storage. (The amount of data actually moved is equal to the number of records multiplied by the physical record length, not the logical record length.) Valid values range from 1 through 32767 records. The NBRRCDS parameter is valid for sequential or random processing and is specified only when the data records are physically located in auxiliary storage in the sequence in which they are processed. This parameter overrides the number of records value specified in the program, and/or in other previously issued OVRDBF commands.

EOFDLY

Specifies the number of seconds to delay when the end-of-file is reached before trying to retrieve additional records. This delay allows other jobs to add records to the file, and have the new records processed without having to restart the job. When the delay time ends, the job is made active, and the database determines whether new records were added. If no new records were added, the job waits for another time delay without informing the application program. When a number of seconds is specified, no end-of-file condition occurs on the given database file until an End Job (ENDJOB) command or forced end of data (FEOD) occurs.

Note: This parameter cannot be specified if MBR(*ALL) was specified on a previously issued OVRDBF command that is still in effect for this file.

There are several ways to end a job that is waiting for records due to an EOFDLY. They are:

- Write a record to the specified file which is recognized by the application program as a last record. The application program may then do a force end of data (FEOD) to start the end-of-file processing or close the file.
- End the job using the controlled value (ENDJOB OPTION(*CNTRLD)) with a delay time greater than the time specified on the EOFDLY time. The DELAY parameter time specified must allow for the EOFDLY time to run out, plus time to process any new records that may have been added to the file, and any end-of-file processing that is done in the user's application. The end-of-file is set by database, and a normal end-of-file condition occurs after new records are retrieved.
- End the job immediately (ENDJOB OPTION(*IMMED)).
- If the job is interactive, start a system request and end the previous request.

***NONE:** Normal end-of-file processing is done.

number-of-seconds: Specify the number of seconds that the program waits between each attempt to get a record when an end-of-file condition occurs. No end-of-file condition is signaled until end of data is forced, or the job is ended with the *CNTRLD option. Valid values range from 1 through 99999 seconds.

LVLCHK

Specifies whether the record format level identifiers in the program are checked against those in the device file when the file is opened. If so, the record format identifiers in the program must match those in the device file. Because the same record format name can exist in more than one file, each record format is given an internal system identifier when it is created.

Note: This parameter overrides the value specified in the database file, in the program, and/or in other previously issued OVRDBF commands issued in this or the following call level. Level checking cannot be done unless the program contains the record format identifiers. This command cannot override level checking from *NO to *YES.

***NO:** The level identifiers are not checked when the file is opened.

EXPCHK

Specifies whether the expiration date of the named member is checked. This date check is valid only on a physical file member. This parameter overrides the value specified in the program, and/or in other called OVRDBF commands.

***YES:** The expiration date of the physical file member is checked. If the current date is later than the expiration date, an error message is sent to the job, where it is monitored. An escape message is sent to the program.

***NO:** The expiration date is not checked.

INHWRT

Specifies whether the processed records are written, deleted, or changed in the database file. This parameter tests a program without storing the processed records back in the database. This parameter overrides the INHWRT parameter in other previously issued OVRDBF commands.

Note: This parameter cannot be specified if MBR(*ALL) was specified on a previously issued OVRDBF command that is still in effect for this file.

***YES:** Processed records are prevented from being written into the database; they are written only to an output device.

***NO:** All new and changed processed records are written into the database, unless the program is in debug mode with UPDPROD(*NO) specified, and the file is in a production library. In that case, an escape message is sent to the program.

SECURE

Specifies whether this file is safe from the effects of previously called file override commands. If SECURE is not specified, processing occurs as if SECURE(*NO) is specified.

***NO:** This file is not protected from the effects of other file overrides; its values can be overridden by the effects of previously called file override commands.

***YES:** This file is protected from the effects of any file override commands previously called.

OVRSCOPE

Specifies the extent of influence (scope) of the override.

***CALLLVL:** The scope of the override is determined by the current call level. All open operations done at a call level that is the same as or higher than the current call level are influenced by this override.

***JOB:** The scope of the override is the job in which the override occurs.

SHARE

Specifies whether the open data path (ODP) for the database file member is shared with other programs in the routing step. When an ODP is shared, the programs accessing the file share facilities such as the file status and the buffer.

More information on shared database files is in the *Database Guide*.

***NO:** The ODP created by the program with this attribute is not shared with other programs in the routing step. Every time a program opens the file with this attribute, a new ODP to the file is created and activated.

Note: This includes several opens in the same program.

***YES:** The ODP created with this attribute is shared with each program in the routing step that also specifies SHARE(*YES) when it opens the file.

Note: When SHARE(*YES) is specified and control is passed to a program, a read operation in that program retrieves the next input record. A write operation produces the next output record.

OPNSCOPE

Specifies the extent of influence (scope) of the open operation.

***ACTGRPDFN:** The scope of the open operation is determined by the activation group of the program that called the OVRDBF command processing program. If the activation group is the default activation group, the scope is the call level of the caller. If the activation group is a non-default activation group, the scope is the activation group of the caller.

***JOB:** The scope of the open operation is the job in which the open operation occurs.

SEQONLY

Specifies, for database files whose records are processed in sequential order only, whether sequential-only processing is used on the file. This parameter also specifies the number of records transferred as a group to or from the database, if sequential-only processing is used. If a number is not specified, a default number is determined by the system. This parameter is used to improve the performance of programs that process database files in a sequential manner. This parameter overrides the value specified in the program and/or in other previously issued OVRDBF commands.

For files opened for *input* only in a program, the specified number of records is transferred as a group from the database to an internal data management buffer.

For files opened for *output* only in a program, a group of records is transferred to the database whenever the internal data management buffer receives the specified number of processed records from the program. For output files, sequential-only processing is valid for physical file members and for logical file members that are based on one physical file member only.

If SEQONLY(*YES) is specified, and any of the following conditions are true, the SEQONLY parameter is ignored and a message is issued.

- The program opened the member for output only and SEQONLY(*YES) is specified with the default number of records, and the member opened is either a logical member, a unique keyed physical member, or other access paths are built over the physical member.
- The program opened the member for other than input or output.
- The member opened by the program for output is based on many other members.
- The record length plus the feedback area sum exceeded 32,767 bytes.

Note: Unpredictable results occur when this parameter is used for alternate index files for DDM on a system other than an AS/400 system.

***NO:** The database file is not restricted to sequential-only processing.

***YES:** The database file uses sequential-only processing. A default value for the number of records transferred as a group is determined by the system based on how the file is used, the type of access path involved, and the file's record length:

- The default is approximately the number of records that fit in an internal buffer of 4K for:
  - All database files opened for input only
  - Physical files opened for output that are only processed in either arrival sequence or in non-unique keyed sequence and that have no logical file members based on them
- The default is 1 record for:
  - All logical files opened for output only

- Physical files opened for output only that either have *unique* keyed sequence access paths or have at least one dependent logical file with a keyed sequence access path that does not share the access path of the keyed physical file member

number-of-records: Specify *YES followed by a value (ranging from 1 through 32767) for the number of records transferred between the database and the internal buffer. The user must ensure that the buffer size specified is always available to the program in the storage pool in which the program is running. The file uses sequential-only processing.

While records are in the internal data management buffer, other jobs can make changes to the same records in the database, and the program performing sequential-only input processing does not see the updates. To ensure that no other updating is done to records while they are in the buffer, the Allocate Object (ALCOBJ) command can be used in the program to specify either an *EXCLRD or an *EXCL lock on the file.

If a program performs sequential-only output processing and does not handle output errors (such as duplicate keys and conversion mapping errors) that may occur when the records in the buffer are written to the database, records in the buffer after the first record in error are not written.

If the file is opened for output and the value specified in this parameter is not the same as the force write ratio specified for the file, the value used by the system is the smaller of the two; a message stating which value is changed is sent to the user.

When processing SEQONLY(*YES) for writing records into a database file, feedback information for each record (such as relative record number) is not always changed. If such feedback information is important, specify SEQONLY(*NO) or SEQONLY(*YES 1).

More information on sequence-only database files is in the *Database Guide*.

Examples

Example 1: Overriding An Existing Member

```
OVRDBF   FILE(ORDERSIN)  MBR(MONDAY)
```

This command overrides the existing member with member MONDAY. With the override in effect, the member MONDAY will be processed when the file ORDERSIN is opened.

Example 2: Overriding a Share Specification

```
OVRDBF   FILE(ORDERSIN)  SHARE(*YES)
```

This command overrides the share specification for the file ORDERSIN. Because of this override, any subsequent opens of this file within the routing step share the ODP for the file.

Example 3: Overriding a File, Member and Lock State

```
OVRDBF   FILE(INPUT)  TOFILE(PAYROLL)  MBR(MBR1)
  RCDFMTLCK((EMPDATA *EXCL))
```

This command overrides the file, the member, and the lock state of the record format EMPDATA. The override will cause the following to occur when the file INPUT is opened:

- The file PAYROLL will be processed instead of the file INPUT.
- The member MBR1 will be processed instead of the previously specified member.
- The lock *EXCL will be placed on record format EMPDATA instead of the existing lock. (*EXCL prevents another program from using the record format while the override is in effect.)

Additional Considerations

The following characteristics apply to the processing of DDM files on the OVRDBF command.

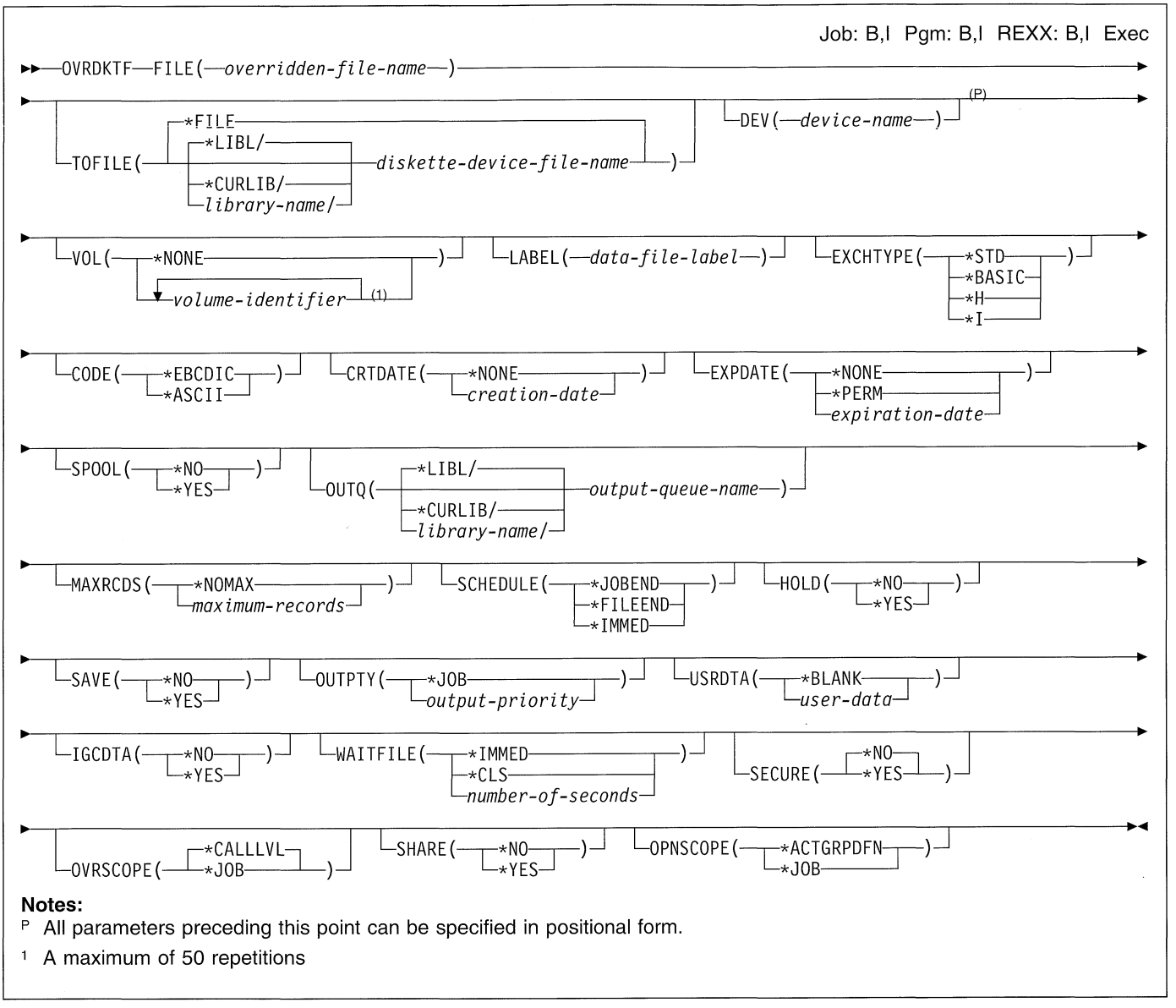
- All parameters are processed normally when the target system is an AS/400 system.
- When the target system is not an AS/400 system:
  - The following parameters are still valid:

EXPCHK	POSITION	SEQONLY	WAITFILE
INHWRT	RCDFMTLCK	SHARE	WAITRCD
LVLCHK	SECURE	TOFILE	

  - The FMTSLR parameter, if specified, causes an error when the file opened is a DDM file.
  - The FRCRATIO and NBRRCDS parameters, if specified, are ignored.
  - The RCDFMTLCK parameter, if specified, is valid only if both of the following are true of the remote file used: (1) Only one type of lock state is requested for the remote file, and (2) the record format name in the remote file must be the same as the name of the distributed data management file.
  - The TOFILE parameter is always processed on the source system. When a DDM file name is specified on this parameter, the program uses the associated remote file instead of the local database file specified in the program.
  - The WAITFILE and WAITRCD parameters have no effect on remote file processing.

The *DDM Guide* has examples of how file overrides are applied in DDM.

OVRDKTF (Override with Diskette File) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 A maximum of 50 repetitions

Purpose

The Override with Diskette File (OVRDKTF) command is used to (1) override (replace) the file named in the program, (2) override certain parameters of a file that is used by the program, or (3) override the file named in the program and override certain parameters of the file processed.

Parameters overridden by this command are specified in the file description, in the program, and/or in other called file override commands. If a file named in the program is overridden, the name of that file is specified in the FILE parameter and the name of the overriding file (the file processed) is specified in the TOFILE parameter. The OVRDKTF command also specifies parameters to override values contained in the file description of the overriding file. If the file named in the program is not replaced but certain parameters of the file are overridden, the name of the file is specified in the FILE parameter and *FILE is specified in the TOFILE parameter. The parameters overridden are then specified by the other parameters of the OVRDKTF command. Parameters that are not specified do not affect parameters specified in the file description, in the program, and/or in other called file override commands.

More information on overriding files is in the *Data Management Guide*, the *Guide to Programming Displays*, and the *Guide to Programming for Printing*.

Required Parameter

FILE

Specifies the name of the file being used by the program to which this override command is applied. If

TOFILE(*FILE) is specified, a display device file must be specified. Otherwise, any device file or database file can be specified.

Optional Parameters

TOFILE

Specifies the qualified name of the diskette file that is used instead of the file specified in the FILE parameter or, if *FILE is specified, specifies that certain attributes are overridden by parameters specified in this command. The parameters specified in this OVRDKTF command override the same parameters specified in the diskette device file in the program, and/or in other called OVRDKTF commands.

***FILE:** The diskette device file named in the FILE parameter has some of its parameters overridden by values specified in this command.

The name of the diskette file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

diskette-device-file-name: Specify the name of the diskette device file that is used instead of the overridden file.

DEV

Specifies the name of the diskette device used with this diskette device file to perform input/output data operations. The device name of the IBM-supplied diskette device description is QDKT. This parameter is ignored if SPOOL(*YES) is specified for the file when it is opened.

This parameter overrides the value specified in the device file, in the program, and/or in other called OVRDKTF commands.

device-name: Specify the name of the device that is used with this diskette device file. The device name must already exist on the system as a device description before this device file is created.

VOL

Specifies one or more volume identifiers used by the file. The volumes must be installed in the same order as the identifiers are specified here (and as they are specified on the DEV parameter). If the file is opened for read backward, then the volume identifiers in the list are processed from last to first (while the devices in the device list are used in first-to-last order). If a list of volume identifiers is provided for the file, operator messages indicate the name of the required volume. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

This parameter overrides the volume identifiers specified in the diskette device file, in the program, and/or in other called OVRDKTF commands.

***NONE:** The diskette volume identifiers are not specified for this file in this command. They can be specified later before the device file is opened, either in a Override with Diskette File (OVRDKTF) command or a Change Diskette File (CHGDKTF) command, or in the high-level language program. Otherwise, no volume identifier checking is done.

volume-identifier: Specify the identifiers of one or more volumes in the order in which they are put on the device and used. Each volume identifier contains up to 6 alphanumeric characters. A blank is used as a separator character when listing multiple identifiers.

LABEL

Specifies the data file label of the data file on diskette that is used with this diskette device file. For input files (diskette input to system), this label specifies the identifier of the file that exists on the diskette. For output files (system output to diskette), the label specifies the identifier of the file that is created on the diskette. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

This parameter overrides the label specified in the diskette device file, in the program, and/or in other called OVRDKTF commands.

data-file-label: Specify up to 8 characters for the identifier of the data file used with this diskette device file.

EXCHTYPE

Specifies, for diskette output files only, the exchange type used by the device file when the system is writing diskette data. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

This parameter overrides the value specified in the device file, in the program, and/or in other called OVRDKTF commands.

***STD:** The basic exchange format is used for a type 1 or a type 2 diskette. The H exchange type is used for a type 2D diskette.

***BASIC:** The basic exchange type is used.

***H:** The H exchange type is used.

***I:** The I exchange type is used.

CODE

Specifies the character code used. The code can be either extended binary-coded decimal interchange code (*EBCDIC) or the American National Standard Code for Information Interchange (*ASCII).

This parameter overrides the value specified in the device file, in the program, and/or in other called OVRDKTF commands.

***EBCDIC:** The extended binary-coded decimal interchange code (EBCDIC) character set code is used.

***ASCII:** The ASCII character set code is used.

CRTDATE

Specifies the date when the diskette data file was created on the diskette.

Note: The creation date parameter is valid only for diskette input data files. If the creation date written on the diskette containing the data file does not match the date specified for the device file when it is opened, an error message is sent to the user program.

This parameter overrides the values specified on the device file, in the program, and/or on other called OVRDKTF commands.

***NONE:** The creation date is not specified. It is not checked unless it is supplied before the device file is opened, either in a OVRTAPF command or CHGTAPF command, or in the high-level language program.

creation-date: Specify the creation date of the data file used by this device file. The date must be specified in the format defined by the job attributes DATFMT and, if separators are used, DATSEP. However, the specified date is put in the diskette label in the format `yymmdd`.

EXPDATE

Specifies the expiration date. The files cannot be overwritten until the expiration date. The expiration date must be later than or equal to the current date.

Note: The expiration date overrides the value specified in the device file, in the program, and/or in other called OVRDKTF commands.

***NONE:** No expiration date for the data file is specified; the file is protected for 1 day. Its protection ends the day after it is created.

***PERM:** The data file is permanently protected. An expiration date of 999999 is assigned.

expiration-date: Specify the expiration date of the data file. The date must be specified in the format defined by the job attributes DATFMT and, if separators are used, DATSEP. However, the specified date is put in the diskette label as `yymmdd`.

SPOOL

Specifies whether the input or output data for the diskette device file is spooled.

This parameter overrides the spool value specified in the device file, and/or in other called OVRDKTF commands.

***NO:** The data is not spooled. If this file is opened for input, the data is read directly from the diskette. If this is an output file, the data is written directly to the diskette as it is processed by the program.

Note: If SPOOL(*NO) is specified, the following parameters in this command are ignored: OUTQ, MAXRCDS, SCHEDULE, HOLD, SAVE, OUTPTY, and USRDTA.

***YES:** The data is spooled. If this file is opened for input, an inline data file having the specified name is processed; otherwise, the next unnamed inline spooled file is processed. More information on named and unnamed inline files is in the *Guide to Programming for Tape and Diskette.* If this is an output file, the data is spooled for processing by a diskette or print writer.

OUTQ

Specifies the qualified name of the output queue used for spooled printer files that specify OUTQ(*JOB). This change does not affect files already created in active jobs or files in completed jobs in which the files were spooled.

This parameter overrides the output queue name specified in the device file, and/or in other called OVRDKTF commands.

The name of the output queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

output-queue-name: Specify the name of the output queue to which the output data is spooled. The IBM-supplied output queue that is used by the diskette file is the QDKT output queue, stored in the QGPL library.

MAXRCDS

Specifies, for spooled files only, the maximum number of records in the spooled file for spooled jobs using this diskette device file.

This parameter overrides the value specified in the diskette device file, and/or in other called OVRDKTF commands.

***NOMAX:** The system maximum is used.

maximum-records: Specify the maximum number of diskette records that are in the spooled file. Valid values range from 1 through 500000.

SCHEDULE

Specifies, for spooled output only, when the spooled file is available to a writer.

This parameter overrides the scheduling value specified in the device file, and/or in other called OVRDKTF commands.

***JOBEND:** The spooled file is made available to the writer only after the entire job is completed.

***FILEEND:** The spooled file is made available to the writer as soon as the file is closed in the program.

***IMMED:** The spooled file is made available to the writer as soon as the file is opened in the program.

HOLD

Specifies, for spooled output only, whether the spooled file is held. The spooled file can be released by using the Release Spooled File (RLSSPLF) command.

Note: This parameter overrides the hold value specified in the diskette device file, and/or in other called OVRDKTF commands.

***NO:** The spooled printer file is not held by the output queue. The spooled output is available to a writer based on the SCHEDULE parameter value.

***YES:** The spooled file is held until released by the Release Spool File (RLSSPLF) command.

SAVE

Specifies, for spooled output only, whether the spooled file is saved (left on the output queue) after the output has been produced.

This parameter overrides the save value specified in the device file, and/or in other called OVRDKTF commands.

Note: This parameter overrides the save value specified in the diskette device file, and/or in other called OVRDKTF commands.

***NO:** The spooled file data is not saved on the output queue after it has been produced.

***YES:** The spooled file data is saved on the output queue until the file is deleted.

OUTPTY

Specifies the output priority for spooled output files that are produced by this job. The highest priority is 1 and the lowest priority is 9. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***JOB:** The output priority associated with the job that created the spooled file is used.

output-priority: Specify the output priority. Valid values range from 1 (high priority) through 9 (low priority).

USRDTA

Specifies, for spooled output only, the user-specified data that identifies the file.

***BLANK:** Ten blanks are used as the user data.

user-data: Specify up to 10 characters of text.

IGCDTA

Specifies whether the file processes double-byte character set (DBCS) data.

***NO:** The file does not process DBCS data.

***YES:** The file processes DBCS data.

WAITFILE

Specifies the number of seconds that the program waits for the file resources and session resources to be allocated when the file is opened, or for the device or session resources to be allocated when an acquire operation is performed to the file. If those resources are not allocated within the specified wait time, an error message is sent to the program. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: An immediate allocation of the device by the device resource is required when an acquire operation is performed to the file.

This parameter overrides the wait time specified in the device file, in the program, and/or in other called OVRDKTF commands.

***IMMED:** The program does not wait; when the file is opened, an immediate allocation of the file resources is required.

***CLS:** The job default wait time is used as the wait time for the file resources being allocated.

number-of-seconds: Specify the number of seconds that the program waits for the file resources to be allocated to the diskette file when the file is opened, or the wait time for the device allocated when an acquire operation is performed to the file. Valid values range from 1 through 32767 seconds.

SECURE

Specifies whether this file is safe from the effects of previously called file override commands. If SECURE is not specified, processing occurs as if SECURE(*NO) is specified.

***NO:** This file is not protected from the effects of other file overrides; its values can be overridden by the effects of previously called file override commands.

***YES:** This file is protected from the effects of any file override commands previously called.

OVRSCOPE

Specifies the extent of influence (scope) of the override.

***CALLLVL:** The scope of the override is determined by the current call level. All open operations done at a call level that is the same as or higher than the current call level are influenced by this override.

***JOB:** The scope of the override is the job in which the override occurs.

SHARE

Specifies whether the open data path (ODP) for the diskette file is shared with other programs in the routing step. When an ODP is shared, the programs accessing the file share facilities such as the file status and the buffer.

More information on shared database files is in the *Database Guide*.

This parameter also overrides the value specified in other called OVRDKTF commands.

***NO:** The ODP created by the program with this attribute is not shared with other programs in the routing

step. Every time a program opens the file with this attribute, a new ODP to the file is created and activated.

***YES:** The ODP created with this attribute is shared with each program in the routing step that also specifies SHARE(*YES) when it opens the file.

Note: When SHARE(*YES) is specified and control is passed to a program, a read operation in that program retrieves the next input record. A write operation produces the next output record.

OPNSCOPE

Specifies the extent of influence (scope) of the open operation.

***ACTGRPDFN:** The scope of the open operation is determined by the activation group of the program that called the OVRDKTF command processing program. If the activation group is the default activation group, the scope is the call level of the caller. If the activation group is a non-default activation group, the scope is the activation group of the caller.

***JOB:** The scope of the open operation is the job in which the open operation occurs.

Examples

Example 1: Changing Spooling Specifications

```
OVRDKTF  FILE(OUT)  VOL(DPT706)  LABEL(STATUSR)
  SPOOL(*YES)
```

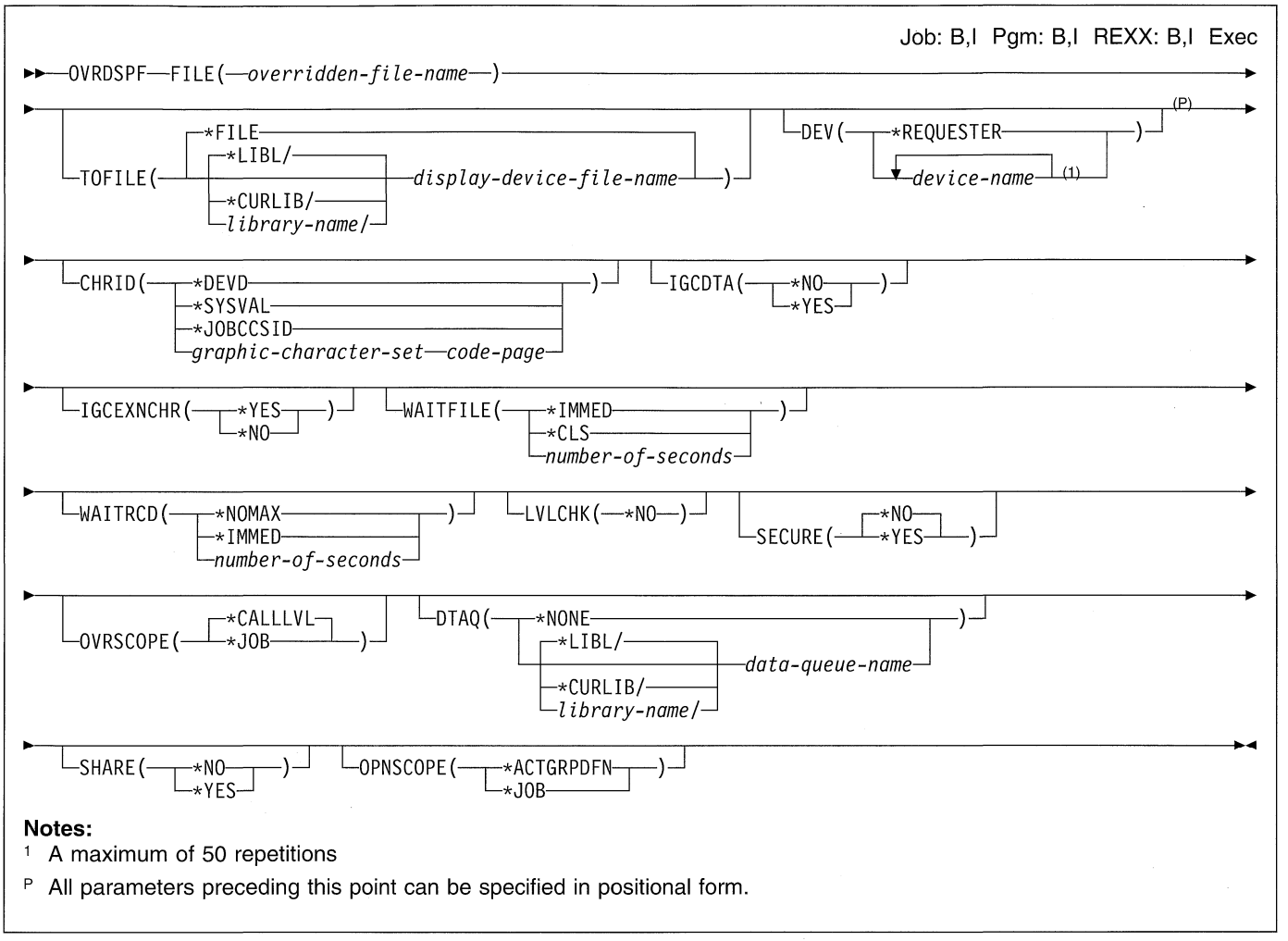
This command changes the spooling specification for the output file named OUT. When a program produces output data for the OUT file, the data is spooled for processing by a spooling writer. The writer processes the data by writing it in a data file called STATUSR that is on a diskette whose volume identifier is DPT706.

Example 2: Specifying DBCS Processing

```
OVRDKTF  FILE(IGCLIB/IGCDCT)  IGCDTA(*YES)
```

This command overrides the diskette file IGCDCT, which is stored in the library IGCLIB, so that the file contains double-byte character set data.

OVRDSPF (Override with Display File) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Override with Display File (OVRDSPF) command is used to (1) override (replace) the file named in the program, (2) override certain parameters of a file used by the program, or (3) override the file named in the program and override certain parameters of the file processed. Parameters overridden by this command are specified in the file description, in the program, and/or in other called file override commands.

If a file named in the program is overridden, the name of that file is specified in the FILE parameter and the name of the overriding file (the file being processed) is specified in the TOFILE parameter. The OVRDSPF command also specifies parameters to override values contained in the file description of the overriding file. If the file named in the program is not replaced but certain parameters of the file are overridden, the name of the file is specified in the FILE parameter and *FILE is specified in the TOFILE parameter. The parameters overridden are then specified by the other parameters of the OVRDSPF command. Parameters that are not specified do not affect parameters specified in the file description, in the program, and/or in other called file override commands.

More information on override files is in the *Guide to Programming Displays*.

Required Parameter

FILE
Specifies the name of the file being used by the program to which this override command is applied. If TOFILE(*FILE) is specified, a display device file must be specified. Otherwise, any device file or database file can be specified.

Optional Parameters

TOFILE
Specifies the qualified name of the display file used instead of the file specified in the FILE parameter or, if *FILE is specified, specifies that certain attributes are overridden by parameters specified in this command. The parameters specified on this OVRDSPF command

override the same parameters specified in the display device file, in the program, and/or in other called OVRDSPF commands.

***FILE:** The display device file named in the FILE parameter has some of its parameters overridden by the values specified in this command.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

display-device-file-name: Specify the name of the display device file used instead of the overridden file.

DEV

Specifies the names of one or more display devices used with this display device file to pass data records between the users of the display devices and their jobs. The device name specified in the display device file supplied by IBM is *REQUESTER.

This parameter overrides the device names specified in the device file, in the program, and/or in other called OVRDSPF commands.

***REQUESTER:** The device from which the program is called is assigned to the file when the file is opened.

device-name: Specify the names of one or more display devices used with this device file to pass data records between the users of the devices and the system. Each device name must already be known on the system by a device description before this device file is created. *REQUESTER can be specified as one of the names. Up to 50 names can be specified in this command, but the total number cannot exceed the number specified on the MAXDEV parameter.

CHRID

Specifies the character identifier (graphic character set and code page) that a work station display device supports. When a display file that was created with the CHRID DDS keyword is used with the device, the system converts data sent to and received from the device to ensure that the correct characters are shown and that the correct hexadecimal byte values are returned to the application program. More information about display file CHRID processing and the translation tables that are used to convert data sent to and received from the display are in the *Guide to Programming Displays*.

***DEVD:** The CHRID value specified in the device description of the work station on which the application is running is used. If no CHRID value is specified, the QCHRID system value (for the system on which the application is running) is used. No translation is necessary because the file has the same character identifier as the work station. For a list of valid values, see the CHRID parameter of the Create Device Description Display (CRTDEVDSP) command description.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***JOBCCSID:** The character data is changed from the device CHRID to the CCSID (coded character set identifier) of the job on display file input, and from the CCSID of the job to the device CHRID on display file output.

Note: This value is not allowed if the file was created on a system at an earlier release level than V2R3M0.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values that match the attributes of the display device. Valid values range from 1 through 32767.

Element 2: Code Page

code-page: Specify the code page set values that match the attributes of the display device. Valid values range from 1 through 32767.

IGCDTA

Specifies, for program-described original files, whether the file processes double-byte character set (DBCS) data. For externally described printer files, this parameter specifies DBCS attributes of the file.

***NO:** The file does not process DBCS data.

***YES:** The file processes DBCS data.

IGCEXNCHR

Specifies whether the system processes double-byte character set (DBCS) extension characters.

***YES:** The system processes DBCS extension characters.

***NO:** The system does not process DBCS extension characters; it displays extension characters as the undefined character.

WAITFILE

Specifies the number of seconds that the program waits for the file resources and session resources to be allocated when the file is opened, or for the device or session resources to be allocated when an acquire operation is performed to the file. If those resources are not allocated within the specified wait time, an error message is sent to the program. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: An immediate allocation of the device by the device resource is required when an acquire operation is performed to the file.

This parameter overrides the wait time specified in the device file, in the program, and/or in other called OVRDSPF commands.

***IMMED:** The program does not wait; when the file is opened, an immediate allocation of the file resources is required.

***CLS:** The job default wait time is used as the wait time for the file resources being allocated.

number-of-seconds: Specify the number of seconds that the program waits for the file resources to be allocated to the display device file when the file is opened, or the wait time for the device allocated when an acquire operation is performed to the file. Valid values range from 1 through 32767 seconds.

WAITRCD

Specifies the number of seconds the program waits for the completion of a read-from-invited-device operation to a multiple device file in a high-level language program. Refer to the appropriate high-level language reference manual to determine when a file is treated as a multiple device file. The program performing the read operation waits for input from all invited devices currently accessing the file. If a record is not returned from an invited device in the specified amount of time, a notify message is sent to the program. This parameter has no effect on an input operation directed to a specific device.

Note: This parameter is also used to specify the time (seconds) that a CL program waits to complete a WAIT command. If a record is not returned from any of the devices that should return a record, an escape message is sent to the CL program. More information on the WAITRCD parameter is in the Receive File (RCVF), Send File (SNDF), Send/Receive File (SNDRCVF), and WAIT (Wait) command descriptions.

This parameter overrides the wait record value specified in the device file, in the program, and/or in other called OVRDSPF commands.

***NOMAX:** There is no limit on the time the system waits for the completion of the operation.

***IMMED:** The program does not wait for the read-from-invited-device operation for the completion of the file. A record must be available from an invited program device when the read-from-invited-program-device operation is performed. If a record is not already available when the read-from-invited-program-device operation is performed, a notify message is sent to the program.

number-of-seconds: Specify the number of seconds that the program waits for the completion of the read-from-invited-device operation. Valid values range from 1 through 32767.

LVLCHK

Specifies whether the record format level identifiers in the program are checked against those in the device file when the file is opened. If so, the record format identifiers in the program must match those in the device file. Because the same record format name can exist in more than one file, each record format is given an internal system identifier when it is created.

Note: This parameter overrides the value specified in the device file, in the program, and/or in other called OVRDSPF commands. Level checking cannot be done unless the program contains the record format identifiers. This command cannot override level checking from *NO to *YES.

***NO:** The level identifiers are not checked when the file is opened.

SECURE

Specifies whether this file is safe from the effects of previously called file override commands. If SECURE is not specified, processing occurs as if SECURE(*NO) is specified.

***NO:** This file is not protected from the effects of other file overrides; its values can be overridden by the effects of previously called file override commands.

***YES:** This file is protected from the effects of any file override commands previously called.

OVRSCOPE

Specifies the extent of influence (scope) of the override.

***CALLLVL:** The scope of the override is determined by the current call level. All open operations done at a call level that is the same as or higher than the current call level are influenced by this override.

***JOB:** The scope of the override is the job in which the override occurs.

DTAQ

Specifies the name of the data queue that receives an entry from the system when a data-available event is signaled from an invited display device. The data queue need not exist when the display file is created since the name specified on this parameter is not evaluated until the file is used. More information on the data queue function is in the *CL Programmer's Guide*.

***NONE:** A data queue does not receive an entry from the system.

The name of the data queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

data-queue-name: Specify the name of the data queue that is to receive an entry from the system when the data-available event is signaled.

SHARE

Specifies whether the open data path (ODP) for the display file is shared with other programs in the routing step. When an ODP is shared, the programs accessing the file share facilities such as the file status and the buffer.

More information on shared database files is in the *Database Guide*.

This parameter overrides the value specified in the device file, in the program, and/or in other called OVRDSPF commands.

***NO:** The ODP created by the program with this attribute is not shared with other programs in the routing step. Every time a program opens the file with this attribute, a new ODP to the file is created and activated.

***YES:** The ODP created with this attribute is shared with each program in the routing step that also specifies SHARE(*YES) when it opens the file.

Note: When SHARE(*YES) is specified and control is passed to a program, a read operation in that program retrieves the next input record. A write operation produces the next output record.

OPNSCOPE

Specifies the extent of influence (scope) of the open operation.

***ACTGRPDFN:** The scope of the open operation is determined by the activation group of the program that called the OVRDSPF command processing program. If the activation group is the default activation group, the scope is the call level of the caller. If the activation group is a non-default activation group, the scope is the activation group of the caller.

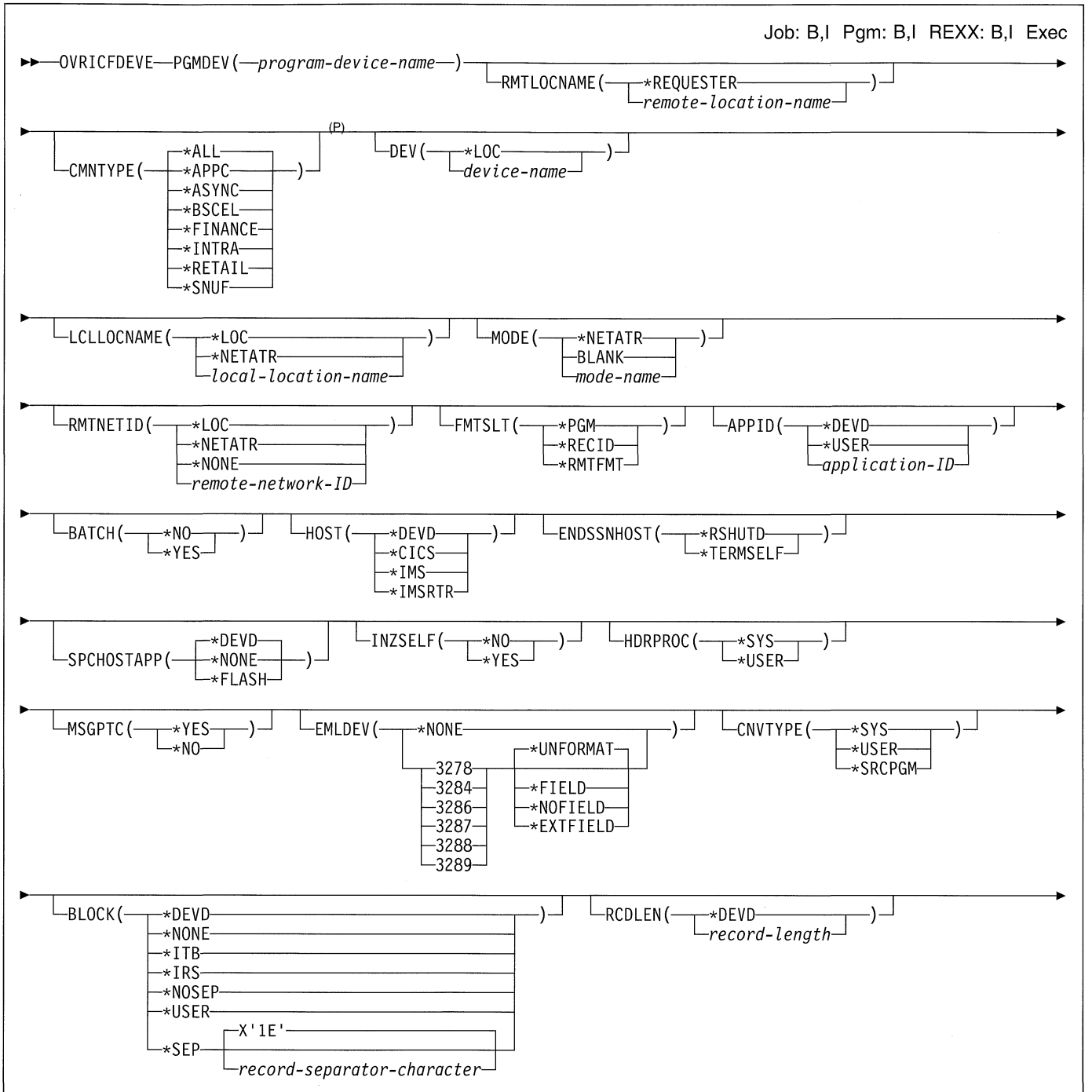
***JOB:** The scope of the open operation is the job in which the open operation occurs.

Example

```
OVRDSPF   FILE(DISPLAY75)  WAITFILE(30)
```

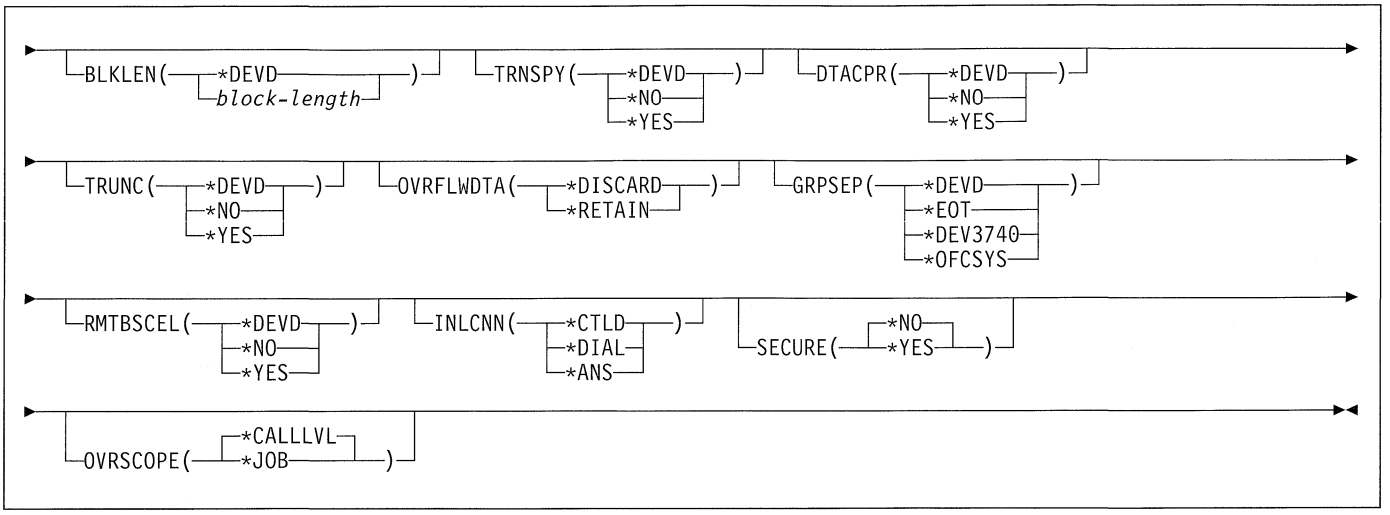
This command overrides the file wait time value specified in the DISPLAY75 device file description, in the program, and/or in other called OVRDSPF commands. The program in which this command occurs waits up to 30 seconds (if necessary) to allocate the required file resources to the file named DISPLAY75.

OVRICFDEVE (Override with Intersystem Communications Function Program Device Entry) Command



Note:

P All parameters preceding this point can be specified in positional form.



Purpose

The Override with Intersystem Communications Function Program Device Entry (OVRICFDEVE) command can be used to temporarily add the program device entry and the remote location name to the ICF file or to override a program device entry with the specified remote location name and attributes for an ICF file.

More information on how overrides processing is performed is in the *Data Management Guide*, the *Guide to Programming Displays*, and the *Guide to Programming for Printing*.

Required Parameter

PGMDEV

Specifies the program device name for an ICF file whose attributes are being overridden. The total number of devices that may be added to an ICF file is determined by the MAXPGMDEV parameter on the Create ICF File (CRTICFF) command or the Change ICF File (CHGICFF) command.

Specify the name of the ICF program device entry with which the program communicates. This name is used in device-specific input/output operations to identify the program device and the session attributes. The program device name must be unique, although the same remote location name may be specified more than once. This allows more than one session to be at the same remote location, and/or to have different attribute values for each session at the same remote location. This program device name must be unique throughout the entries for the ICF file. If an override command is entered a second time for the same program device, then both (according to the override process rules) define the same program device entry.

Optional Parameters

Note: Refer to the *APPC Programmer's Guide* for information on how the system uses the RMTLOCNAME, DEV, LCLLOCNAME, and RMTNETID parameters to select an APPC device description.

RMTLOCNAME

Specifies the remote location name of the system with which this object communicates.

Note: A remote location must be specified by using the Add Intersystem Communications Function Program Device Entry (ADDICFDEVE) command or an applied program device override. If a remote location is not specified, an escape message is sent when the program device is acquired.

***REQUESTER:** The name used to refer to the communications device through which the program is started is used. The session that is assigned when the program device is acquired is the same session in which the program start request is received. If the program is not started as a result of a program start request, the acquire operation of the program device fails. The target program uses *REQUESTER as the remote location name in the intersystem communications function (ICF) file to connect to the session that the source program used to send the program start request.

The *REQUESTER value can be specified on only one program device entry and is valid only for a target communication job. If *REQUESTER is specified in any other type of job, a message is sent.

remote-location-name: Specify the full name of a remote location. The remote location does not need to exist when this command is run, but it must exist (be configured on the system or in the advanced peer-to-peer network (APPN) for this remote location) at the time the program acquires the program device. A given remote location may be added many times by using different program device names. When a program is running, however, only one program device name associated with

each asynchronous or binary synchronous communications equivalence link (BSCEL) or system network architecture upline facility (SNUF) remote location may read or change the file at any one time. For each remote advanced program-to-program communications (APPC), INTRA, or SNUF location, more than one associated program device name may be acquired to the file at one time. Each SNUF remote location may have many devices. The system determines which device to use unless a device is specified on the DEV parameter.

CMNTYPE

Specifies which communications types are used. This parameter is used only for prompting purposes; it is ignored when the command is run. The value specified for this parameter determines the subset of other parameters that are displayed (prompted) for the user.

***ALL:** The parameters for all of the communications types appear in the prompt.

***APPC:** The advanced program-to-program communications (APPC) parameters appear in the prompt.

***ASYNC:** The asynchronous (ASYNC) parameters appear in the prompt.

***BSCEL:** The binary synchronous communications equivalence link (BSCEL) parameters appear in the prompt.

***FINANCE:** The finance parameters appear in the prompt.

***INTRA:** The intrasystem (INTRA) parameters appear in the prompt.

***RETAIL:** The retail parameters appear in the prompt.

***SNUF:** The SNA upline facility (SNUF) parameters appear in the prompt.

DEV

Specifies the communications device used in the remote location. This parameter should be specified only for APPC, INTRA, and SNUF communications types. If the Add Intersystem Communications Function Program Device Entry (ADDICFDEVE) command is not run for the specified program device and this parameter is not overridden, DEV(*LOC) is used.

***LOC:** The device associated with the remote location is used. If several devices are associated with the remote location, the system determines which device is used.

device-name: Specify the name of a communications device associated with the remote location. If the device name is not valid for the remote location, a message is sent when the program device entry is acquired. More information on device names is in the *APPC Programmer's Guide*.

LCLLOCNAME

Specifies the local location name.

Note: This parameter applies to the APPC communications type only and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device or is not overridden, this parameter defaults to *LOC.

***LOC:** The device associated with the remote location is used. If several devices are associated with the remote location, the system determines which device is used.

***NETATR:** The LCLLOCNAME value specified in the system network attributes is used.

local-location-name: Specify the name of the user's location. If the local location name is not valid for the remote location or remote location and device, an escape message is sent when the program device entry is acquired.

MODE

Specifies the mode name used. This parameter applies only to the APPC communications type and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, MODE(*NETATR) is used.

***NETATR:** The mode name specified in the network attributes is used.

***BLANK:** The mode name consisting of 8 blank characters is used.

mode-name: Specify a mode name for the APPC communications device. If the specified mode is not valid for any combination of remote location, device, local location, and remote network ID, an escape message is sent when the program device entry is acquired.

RMTNETID

Specifies the remote network ID used with the remote location. This parameter applies to the APPC communications type only and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, RMTNETID(*LOC) is used.

***LOC:** The remote network identifier (ID) associated with the remote location is used. If several remote network IDs are associated with the remote location, the system determines which remote network ID is used.

***NETATR:** The RMTNETID value specified in the system network attributes is used.

***NONE:** No remote network identifier (ID) is used.

remote-network-ID: Specify a remote network ID for the APPC communications device.

FMTSLT

Specifies the record format selection used for input operations. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, FMTSLT(*PGM) is used.

***PGM:** The program determines which record formats are selected. If an input (read) operation with a record format name is specified, that format is selected. If an input operation without a record format is specified, the default format (the first record format in the file) is selected. This also means that if there are any record identification (RECID) parameters specified in the data description specifications (DDS) for the file, or if any remote formats are received, they are not taken into consideration when the record is selected.

***RECID:** The record identification (RECID) keywords specified in the DDS for the file are used to do record selection. If there are no RECID keywords in the file, an error message is sent, the acquire operation of the program device ends, and the device is not acquired.

***RMTFMT:** The remote format names received from the sending system are used to do record selection. If the device is not an APPC or intrasystem device and *RMTFMT is specified, a run time error occurs at the time the program device entry is acquired.

APPID

Specifies (in characters) the VTAM identifier of the Customer Information Control System for Virtual Storage (CICS/VS) or Information Management System/Virtual Storage (IMS/VS) host subsystem sent with the sign-on message. This parameter applies to the SNUF communications type only and is ignored for all other devices. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, APPID(*DEVD) is used.

***DEVD:** The application identifier specified in the device description is sent with the sign-on message.

***USER:** The application program can send messages or a logon to the host. This is valid only when using the 3270 program interface.

application-ID: Specify the application identifier that is sent with the sign-on message.

BATCH

Specifies, for CICS/VS and IMS/VS, whether this session is used for batch jobs. This parameter applies to the SNUF, Retail, and INTRA communications types only and is ignored for all other devices. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, BATCH(*NO) is used.

***NO:** Batch jobs do not occur.

***YES:** Batch jobs occur and SNUF is not to assemble physical records into logical records. If BATCH(*YES) is specified, MSGPTC(*NO) must also be specified.

HOST

Specifies the host or remote subsystem with which this session communicates. This parameter applies to the SNUF communications type only and is ignored for all other devices. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, HOST(*DEVD) is used.

***DEVD:** The host system specified in the device description is used.

***CICS:** The session communicates with CICS/VS.

***IMS:** The session communicates with IMS/VS.

***IMSRTR:** The session communicates with IMS/VS using the ready-to-receive option.

ENDSSNHOST

Specifies how SNUF ends the session with the host.

Note: This parameter applies to SNUF communications type only and is ignored for all other devices.

***RSHUTD:** SNUF sends a request-shut-down command to the host.

***TERMSELF:** SNUF sends a terminate-self command to the host. It may be necessary to specify this value if the value *RSHUTD fails to end a session with a non-IBM host.

SPCHOSTAPP

Specifies whether SNUF customizes support for special host applications outside the CICS or IMS application layer.

***DEVD:** The special host application specified in the device description is used.

***NONE:** SNUF does not customize support for special host applications.

***FLASH:** SNUF customizes support for the Federal Link Access for Secondary Half-sessions (*FLASH) protocol application.

INZSELF

Specifies whether a formatted INIT-SELF is built in place of the unformatted sign-on normally sent by SNUF to the host.

***NO:** The unformatted default sign-on provided by SNUF is used.

***YES:** The formatted INIT-SELF provided by SNUF is used.

HDRPROC

Specifies, for both CICS/VS and IMS/VS, whether received function management headers are passed to the application program. This parameter applies to the SNUF communications type only and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, HDRPROC(*SYS) is used.

***SYS:** SNA upline facility (SNUF) removes function management headers before passing data to the program.

***USER:** Function management headers are passed with the data to the program.

MSGPTC

Specifies, for both CICS/VS and IMS/VS, whether message protection is used for this session. This parameter applies to the SNUF communications type only and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device, MSGPTC(*YES) is used.

***YES:** Message protection is used. SNUF saves messages until they are responded to and tries resynchronization if errors occur. *YES is only valid when BATCH(*NO) is also specified.

***NO:** Message protection is not used.

EMLDEV

Specifies that the program device entry is used to send and receive data streams to and from specific types of 3270 display or printer devices being emulated. This parameter consists of an emulation device type and an emulation device data format. The emulation device data format specifies the format of the type 3270 data stream being sent or received. A 20- or 32-byte common header that contains type 3270 command and data flow information is located at the start of the I/O buffer that is sending or receiving the type 3270 data stream. This parameter applies to the SNUF communications type only. This parameter can be specified as a list of two values (elements) or as a single value (*NONE).

***NONE:** This program device entry is not used for sending and receiving 3270 data streams.

Element 1: Type of Device

3278: The data stream is for a 3279, 3278 or 3277 display device.

3284: The data stream is for a 3284 Printer.

3286: The data stream is for a 3286 Printer.

3287: The data stream is for a 3287 Printer.

3288: The data stream is for a 3288 Printer.

3289: The data stream is for a 3289 Printer.

Element 2: Format of Data Stream

***UNFORMAT:** An unformatted 3270 data stream is sent or received. The user application program must translate the data stream into a display or printer image.

***FIELD:** A formatted 3270 data stream is sent or received. The formatted 3270 data stream contains a display or printer image followed by field definitions. The field definitions indicate the location and characteristics of each field.

***NOFIELD:** A formatted 3270 data stream that has no field definitions but contains a display or printer image is sent or received.

Note: If *FIELD or *NOFIELD is specified, BATCH(*NO) must also be specified.

***EXTFIELD:** A formatted 3270 data stream contains extended field attribute information. The extended field attribute information is in the field definitions which follow the display image. The field definitions indicate the location and characteristics of each field. This value can only be specified if 3278 is also specified for the type of device on the EMLDEV parameter.

CNVTYPE

Specifies the conversation type for which the application program is designed. This parameter is valid only for the APPC communications type and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, CNVTYPE(*SYS) is used.

More information on the APPC communications type is in the *APPC Programmer's Guide*.

***SYS:** The advanced program-to-program communications (APPC) mapped conversation support is used.

***USER:** The advanced program-to-program communications (APPC) basic conversation support is used.

***SRCPGM:** The target program accepts the conversation type specified by the source program.

BLOCK

Specifies whether the system or the user controls the combination of records into blocks when they are sent. This parameter is for the BSCEL communications type only and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, BLOCK(*DEVD) is used.

With this parameter, specify one of the following conditions of record blocking:

- No blocking/deblocking: The record format described in the DDS is the format for both the record and the block.
- User blocking and deblocking: Give the BSC controls needed to describe the record format of the system.
- System blocking with record separator characters: Specify the record separator character used by the system to determine record boundaries in the block.
- System blocking of fixed-length records: The system uses fixed-length records, and blocks and deblocks records accordingly.

If a parameter value other than *NONE or *USER is specified, records are blocked as required by the system for output and are deblocked on input.

Element 1: Blocking Option

***DEVD:** The block option in the device description is used.

***NONE:** Blocking or deblocking is not done by the system.

***ITB:** The records are blocked or deblocked, based on the location of an Intermediate Text Block (ITB) control character. For input files, a record is delimited by locating the next intermediate text block character. An end-of-text or end-of-transmission block character is used as an intermediate text block character to delimit a block. For output files, an ITB character is added after the record. If it is the last character of the block, the ITB is replaced by an end-of-text or end-of-transmission block character.

***IRS:** The records are blocked or deblocked based on the location of an Interrecord Separator (IRS) character. For input files, a record is delimited by locating the next IRS character. For output files, an IRS character is added after the record.

***NOSEP:** No record separator character is contained in the block that is sent to or received from the device. The system blocks and deblocks the records using a fixed-length record, as specified in the DDS format specifications.

***USER:** The program provides all the control characters (including record separator characters, binary synchronous communications (BSC) framing characters, and transparency characters) necessary to send records. More information about the device and binary synchronous communications equivalence link (BSCEL) support characteristics is in the *BSC Equivalence Link Programmer's Guide*.

***SEP:** Records are blocked or deblocked, based on the location of a user-specified record separator character. For input files, a record is delimited by locating the next record separator character. For output files, a record separator character is added after the record.

Element 2: Record Separator

X'1E': The record separator X'1E' is used.

record-separator-character: Specify a unique 1-byte record separator character. The record separator character may be specified as 2 hexadecimal characters, as in BLOCK(*SEP X'FD'), or as a single character, as in BLOCK(*SEP @). If a record separator character is not specified, the record separator character X'1E' is used.

The following is a list of BSC control characters that should not be used as record separator characters:

EBCDIC	ASCII	BSC Control
X'01'	X'01'	SOH (start-of-header)
X'02'	X'02'	STX (start-of-text)
X'03'	X'03'	ETX (end-of-text)
X'10'	X'10'	DLE (data-link escape)
X'1D'	X'1D'	IGS (interchange group separator)
X'1F'	X'1F'	ITB (intermediate text block)
X'26'	X'17'	ETB (end-of-transmission block)
X'2D'	X'05'	ENQ (enquiry)
X'32'	X'16'	SYN (synchronization)
X'37'	X'04'	EOT (end-of-transmission)
X'3D'	X'15'	NAK (negative acknowledgment)

RCDLEN

Specifies the maximum record length (in bytes) for data sent and received. This parameter applies to the SNUF and the BSCEL communications types only and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, RCDLEN(*DEVD) is used.

***DEVD:** The record length specified in the device description is used. If a record is longer than the specified record length, a run time error occurs at the time the record is sent or received.

record-length: Specify the maximum record length (in bytes) to use with this device file. The value must be at least the size of the largest record sent. If a record is longer than the specified record length, a run time error occurs when the record is sent or received. Valid values range from 1 through 32767 bytes for SNUF communications. For BSCEL communications, the maximum record length is 8192 bytes.

BLKLEN

Specifies the maximum block length (in bytes) for data sent. This parameter applies to the BSCEL and the SNUF communications types and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, this parameter defaults to *DEVD.

***DEVD:** The block length specified in the device description is used.

block-length: Specify the maximum block length (in bytes) of records sent. The value must be at least the size of the largest record sent. Valid values range from 1 through 32767 for SNA upline facility (SNUF). For binary synchronous communications equivalence link (BSCEL) communications, the maximum block length is 8192.

TRNSPY

Specifies whether text is sent in transparent text mode. Transparent text mode sends all 256 extended binary-coded decimal interchange code (EBCDIC) character codes; use this feature when sending packed or binary data fields. This parameter is for the BSCEL communications type only and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, TRNSPY(*DEVD) is used.

***DEVD:** The text transparency option specified in the device description is used.

***NO:** Text transparency is not used.

***YES:** Text transparency is used, which sends all 256 EBCDIC character codes. *YES is valid only when BLOCK(*NONE), BLOCK(*NOSEP), or BLOCK(*USER) is specified.

Note: Transparency of received data is determined by the data stream; therefore, this parameter is not relevant for received data. If TRNSPY(*YES) is specified with BLOCK(*USER), BSCEL ignores the transparency indicator during write operations. Correct controls must be given with the data to get transparent transmission of data. For example, first specify the DLE and STX control characters; the system gives the remaining control characters for transparent transmission of data.

DTACPR

Specifies whether data compression is performed.

Note: This parameter is for the BSCEL communications type only and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, DTACPR(*DEVD) is used.

***DEVD:** The data compression option specified in the device description is used.

***NO:** No data compression or decompression occurs.

***YES:** Data is compressed for output and decompressed for input. DTACPR(*YES) cannot be specified if TRNSPY(*YES) is specified.

TRUNC

Specifies whether trailing blanks are removed from any output records. TRUNC(*YES) cannot be specified if BLOCK(*NOSEP) or BLOCK(*ITB) is specified. If TRUNC(*YES) is specified when DTACPR(*YES) or BLOCK(*USER) is specified, truncation is ignored. This parameter is for BSCEL communications types only and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, TRUNC(*DEVD) is used.

***DEVD:** The trailing blanks specified in the device description are used.

***NO:** Trailing blanks are not removed from output records.

***YES:** Trailing blanks are removed from output records.

OVRFLWDTA

Specifies whether overflow data is discarded or retained.

***DISCARD:** Overflow data is not kept.

***RETAIN:** Overflow data is kept.

GRPSEP

Specifies a separator for groups of data (for example, data sets and documents). This parameter is for the BSCEL communications type only and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, GRPSEP(*DEVD) is used.

***DEVD:** The group separator option specified in the device description is used.

***EOT:** The end-of-transmission (EOT) BSC control character is used as a data group separator.

***DEV3740:** A null record (STXETX) is used as a data group separator.

***OFCSYS:** A block sent with the end-of-text (ETX) BSC control character is used as a data group separator.

RMTBSCEL

Specifies whether the type of BSCEL session is with a BSCEL system. This parameter applies to the BSCEL communications type only and is ignored for all other communications types. If the ADDICFDEVE command is not run for the specified program device and this parameter is not overridden, RMTBSCEL(*DEVD) is used.

***DEVD:** The RMTBSCEL option specified in the device description is used.

***NO:** A RMTBSCEL attribute of *NO indicates that the remote system cannot recognize BSCEL commands or messages. In most cases, *NO is used when communicating with remote systems such as a 3741 Data Entry Station, an Office System 6, a 5230 Data Collection System, or a System/38.

***YES:** The remote system recognizes the BSCEL transaction starting commands, transaction ending commands, and online messages. In most cases, *YES indicates that the remote system is another AS/400 system, System/38, System/36, or a System/34 with BSCEL support.

INLCNN

Specifies the method used to make a connection on the line for the session being acquired. This parameter applies to the binary synchronous communications equivalence link (BSCEL) communications types only.

***CTLD:** The initial connection option specified in the controller description is used.

***DIAL:** The local system starts the call.

***ANS:** The remote system starts the call and the local system answers the call.

SECURE

Specifies whether this file is safe from the effects of previously called file override commands. If SECURE is not specified, processing occurs as if SECURE(*NO) is specified.

***NO:** This file is not protected from the effects of other file overrides; its values can be overridden by the effects of previously called file override commands.

***YES:** This file is protected from the effects of any file override commands previously called.

OVRSCOPE
Specifies the extent of influence (scope) of the override.

***CALLLVL:** The scope of the override is determined by the current call level. All open operations done at a call level that is the same as or higher than the current call level are influenced by this override.

***JOB:** The scope of the override is the job in which the override occurs.

Examples

Example 1: Overriding the Device Entry with the Record Format Selection Attributes

```
OVRICFDEVE  PGMDEV(BSCEL2)  RMTLOCNAME(BSCNYC)
  FMTSLT(*RECID)
```

This command overrides the program device named BSCEL2 with a corresponding remote location named BSCNYC for any ICF file associated with the job. The program device is overridden with the attributes of FMTSLT(*RECID).

Example 2: Overriding the Device Entry with the Record Format Selection and the Conversation Type Attributes

```
OVRICFDEVE  PGMDEV(APPC1)  RMTLOCNAME(*REQUESTER)
  FMTSLT(*RMTFMT)  CNVTYPE(*SYS)
```

This command overrides the program device entry named APPC1 with a remote location name of *REQUESTER. The program device entry is overridden with the FMTSLT(*RMTFMT) and CNVTYPE(*SYS) attributes.

Example 3: Overriding an Entry for Associated ICF Files

```
OVRICFDEVE  PGMDEV(JOE)  RMTLOCNAME(LU0MPLS)
```

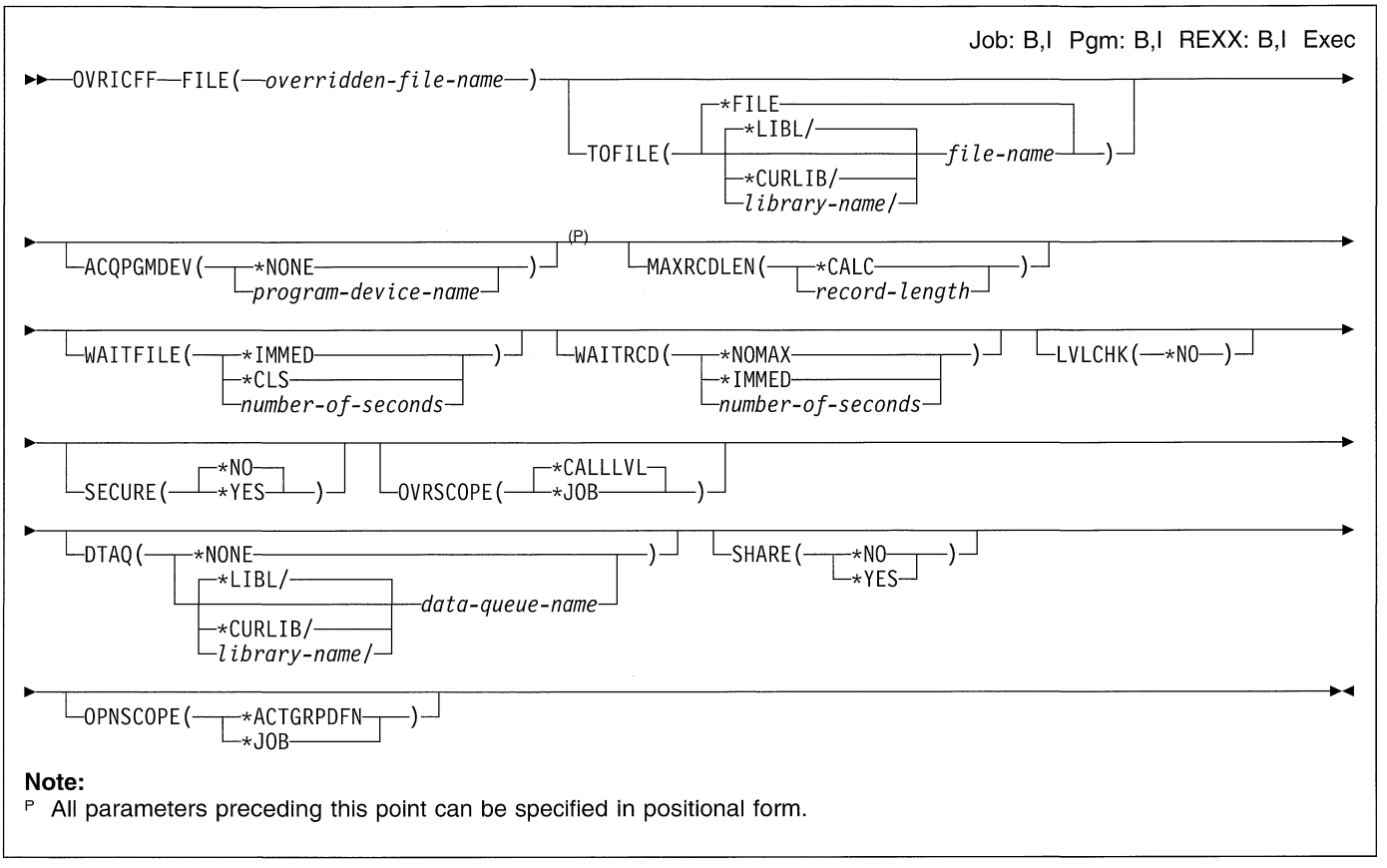
This command overrides the program device entry named JOE with a remote location named LU0MPLS for any ICF file associated with the job.

Example 4: Specifying the Communications Device

```
OVRICFDEVE  PGMDEV(APPC)  RMTLOCNAME(APPCMPLS)
  DEV(MPLSLINE2)
```

This command overrides the program device entry named APPC with a remote location named APPCMPLS using device MPLSLINE2.

OVRICFF (Override with Intersystem Communications Function File) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Override with Intersystem Communications Function File (OVRICFF) command overrides the file named in the program and overrides certain parameters of the file being processed. Parameters overridden by this command can be specified in the file description, in the program, and/or in other file override commands running later.

If a file named in the program is being overridden, the name of that file is specified in the FILE parameter and the name of the overriding file (the file being processed) is specified in the TOFILE parameter.

This command can also specify parameters to override values contained in the file description of the overriding file. If the file named in the program is not being replaced but certain parameters of the file are being overridden, the name of the file is specified in the FILE parameter and *FILE is specified in the TOFILE parameter. The parameters being overridden are then specified by the other parameters of the OVRICFF command. Parameters that are not specified do not affect parameters specified in the file description, in the program, and/or in other override commands run later.

More information on how overrides processing is performed is in the *Data Management Guide*, the *Guide to Programming Displays*, and the *Guide to Programming for Printing*.

Required Parameter

FILE

Specifies the name of the file being used by the program to which this override command is applied. If TOFILE(*FILE) is specified, a database file must be specified. Otherwise, any device file or database file can be specified.

Optional Parameters

TOFILE

Specifies the qualified name of the ICF file (up to 10 characters in length) that is used instead of the file specified in the FILE parameter or, if *FILE is specified, specifies that certain attributes are overridden by parameters specified in this command. The parameters specified in this command override the other values specified in the ICF file or in the program.

***FILE:** Some of the parameters of the ICF file named in the FILE parameter are overridden by values specified in this parameter.

The name of the ICF file description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the qualified name of the physical file or device file that receives the copied records. If no library qualifier is specified, *LIBL is used to locate the file. However, if CRTFILE(*YES) is specified and the specified file cannot be found, the file name must be qualified with a library name. When the physical to-file is created, it is placed in the specified library.

ACQPGMDEV

Specifies which program device is acquired to the file when the file is opened. This parameter overrides the value in the ICF file, in the program, or in the other OVRICFF commands run later.

***NONE:** The file is opened without any devices acquired. All devices used with this file must be explicitly acquired before input/output is directed to them.

program-device-name: Specify the name of the program device that is acquired when the file is opened. The name should be specified on the Add Intersystem Communications Function Program Device Entry (ADDICFDEVE) command or the Override Intersystem Communications Function Program Device Entry (OVRICFDEVE) command as a program-device-name before the file is opened.

MAXRCDLEN

Specifies the maximum record length used when the file is opened. This parameter overrides the value in the ICF file, in the program, and/or in the other OVRICFF commands run later.

***CALC:** The value calculated in the file is used when the file is opened.

record-length: Specify the maximum record length (in bytes) used when the file is opened. Valid values range from 1 through 32767.

WAITFILE

Specifies the number of seconds that the program waits for the file resources and session resources to be allocated when the file is opened, or for the device or session resources to be allocated when an acquire operation is performed to the file. If those resources are not allocated within the specified wait time, an error message is sent to the program. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: An immediate allocation of the device by the device resource is required when an acquire operation is performed to the file.

This parameter overrides the wait time specified in the device file, in the program, and/or in other OVRICFF commands run later.

***IMMED:** The program does not wait; when the file is opened, an immediate allocation of the file resources is required.

***CLS:** The job default wait time is used as the wait time for the file resources being allocated.

number-of-seconds: Specify the number of seconds that the program waits for the file resources to be allocated to the ICF file when the file is opened, or the wait time for the device allocated when an acquire operation is performed to the file. Valid values range from 1 through 32767 seconds.

WAITRCD

Specifies the number of seconds the program waits for the completion of a read-from-invited-device operation to a multiple device file in a high-level language program. Refer to the appropriate high-level language reference manual to determine when a file is treated as a multiple device file. The program performing the read operation waits for input from all invited devices currently accessing the file. If a record is not returned from an invited device in the specified amount of time, a notify message is sent to the program. This parameter has no effect on an input operation directed to a specific device.

Note: This parameter is also used to specify the time (seconds) that a CL program waits to complete a WAIT command. If a record is not returned from any of the devices that should return a record, an escape message is sent to the CL program. More information on the WAITRCD parameter is in the Receive File (RCVF), Send File (SNDF), Send/Receive File (SNDRCVF), and WAIT (Wait) command descriptions.

This parameter overrides the wait record value specified in the device file, in the program, and/or in other override commands started later.

***NOMAX:** There is no limit on the time the system waits for the completion of the operation.

***IMMED:** The program does not wait for the read-from-invited-device operation for the completion of the file. A record must be available from an invited program device when the read-from-invited-program-device operation is performed. If a record is not already available when the read-from-invited-program-device operation is performed, a notify message is sent to the program.

number-of-seconds: Specify the number of seconds that the program waits for the completion of the read-from-invited-device operation. Valid values range from 1 through 32767.

LVLCHK

Specifies whether the record format level identifiers in the program are checked against those in the device file when the file is opened. If so, the record format identi-

fiers in the program must match those in the device file. Because the same record format name can exist in more than one file, each record format is given an internal system identifier when it is created.

Note: This parameter overrides the value specified in the device file, in the program, and/or in other override commands run later. Level checking cannot be done unless the program contains the record format identifiers. This command cannot override level checking from *NO to *YES.

***NO:** The level identifiers of the record formats are not checked when the file is opened.

SECURE

Specifies whether this file is safe from the effects of previously called file override commands. If SECURE is not specified, processing occurs as if SECURE(*NO) is specified.

***NO:** This file is not protected from the effects of other file overrides; its values can be overridden by the effects of previously called file override commands.

***YES:** This file is protected from the effects of any file override commands previously called.

OVRSCOPE

Specifies the extent of influence (scope) of the override.

***CALLLVL:** The scope of the override is determined by the current call level. All open operations done at a call level that is the same as or higher than the current call level are influenced by this override.

***JOB:** The scope of the override is the job in which the override occurs.

DTAQ

Specifies the name of the data queue that receives an entry from the system when a data-available event is signaled from an invited display device. The data queue need not exist when the display file is created since the name specified on this parameter is not evaluated until the file is used. More information on the data queue function is in the *CL Programmer's Guide*.

***NONE:** A data queue does not receive an entry from the system.

The name of the data queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

data-queue-name: Specify the name of the data queue that is to receive an entry from the system when the data-available event is signaled.

SHARE

Specifies whether the open data path (ODP) for the ICF file is shared with other programs in the routing step. When an ODP is shared, the programs accessing the file share facilities such as the file status and the buffer.

More information on shared database files is in the *Database Guide*.

***NO:** The ODP created by the program with this attribute is not shared with other programs in the routing step. Every time a program opens the file with this attribute, a new ODP to the file is created and activated.

***YES:** The ODP created with this attribute is shared with each program in the routing step that also specifies SHARE(*YES) when it opens the file.

Note: When SHARE(*YES) is specified and control is passed to a program, a read operation in that program retrieves the next input record. A write operation produces the next output record.

OPNSCOPE

Specifies the extent of influence (scope) of the open operation.

***ACTGRPDFN:** The scope of the open operation is determined by the activation group of the program that called the OVRICFF command processing program. If the activation group is the default activation group, the scope is the call level of the caller. If the activation group is a non-default activation group, the scope is the activation group of the caller.

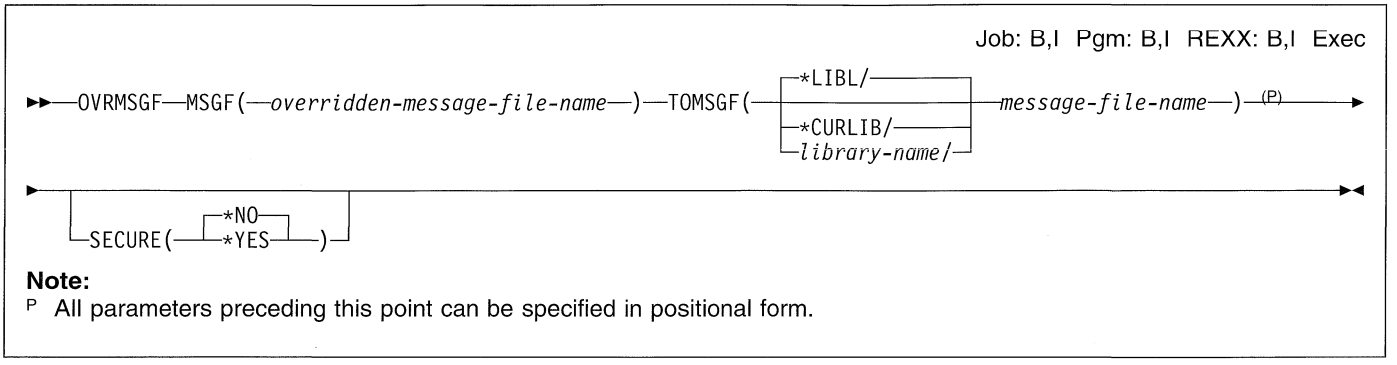
***JOB:** The scope of the open operation is the job in which the open operation occurs.

Example

```
OVRICFF  FILE(ICFHIST)  TOFILE(PRSNNL/ICFCURT)
```

This command overrides the file named ICFHIST to the ICF file named ICFCURT in library PRSNNL.

OVRMSGF (Override with Message File) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Override with Message File (OVRMSGF) command overrides a message file used in a program. The overriding message file (specified in the TOMSGF parameter) is used whenever a message is sent or retrieved and the overridden message file is specified.

The overriding message file need not contain all the messages that the overridden file contains. When a message is received or retrieved and the message identifier cannot be found in the overriding message file, the overridden message file is searched for the identifier. Overriding message files can be overridden, resulting in a chain of overrides. This chain of overrides provides a list of message files that are searched when a message is received or retrieved. Up to 30 message file overrides can be specified in a program.

More information on overriding files is in the *Data Management Guide*, the *Guide to Programming Displays*, and the *Guide to Programming for Printing*.

Required Parameters

MSGF

Specifies the name of the message file being used by the program to which this override command is applied.

TOMSGF

Specifies the qualified name of the message file that is used instead of the message file specified in the MSGF parameter or, if the names are the same, specifies that the SECURE parameter specified in the command is used for the message file.

The name of the message file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-file-name: Specify the name of the message file to use.

Optional Parameters

SECURE

Specifies whether this file is safe from the effects of previously called file override commands. If SECURE is not specified, processing occurs as if SECURE(*NO) is specified.

***NO:** This file is not protected from the effects of other file overrides; its values can be overridden by the effects of previously called file override commands.

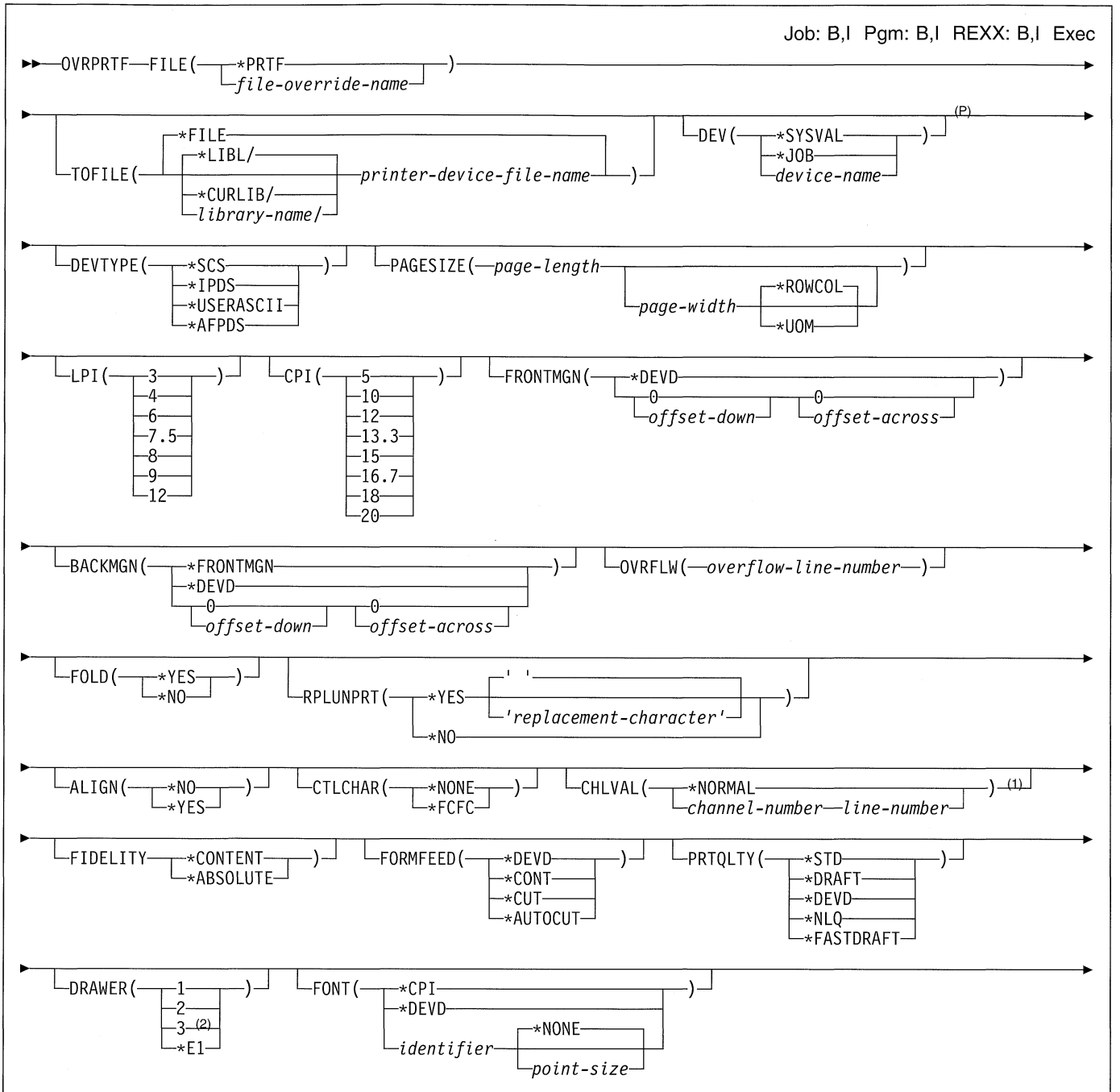
***YES:** This file is protected from the effects of any file override commands previously called.

Example

```
OVRMSGF   MSGF(WSUSRMSG)  TOMSGF(ORDENTMSGD)
```

This override command causes the defaults for messages stored in ORDENTMSGD to be used instead of defaults stored in WSUSRMSG (which contains messages designed for work station users). As a result of this command, the messages received by the order entry users are tailored to their own environment.

OVRPRTF (Override with Printer File) Command

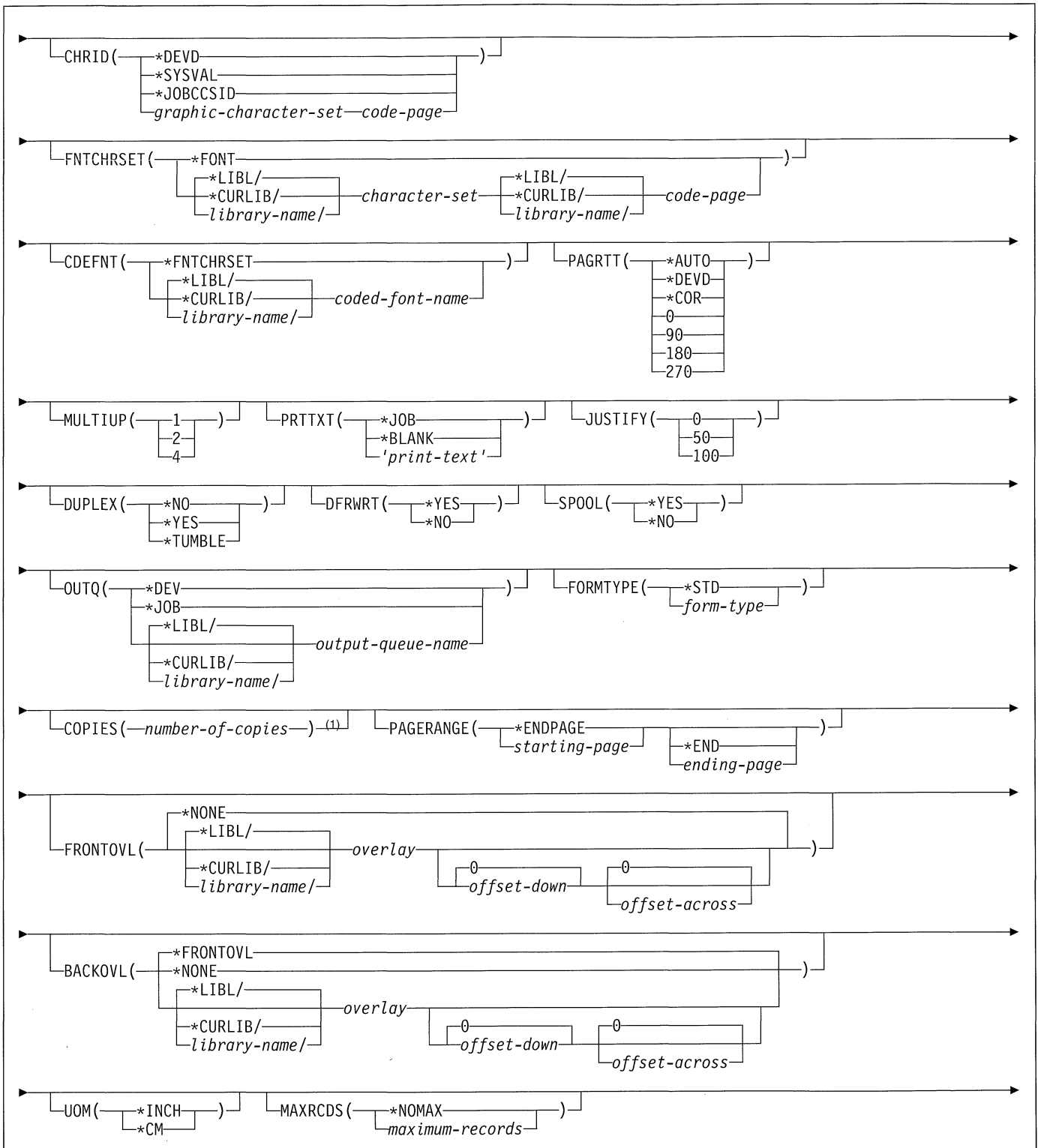


Notes:

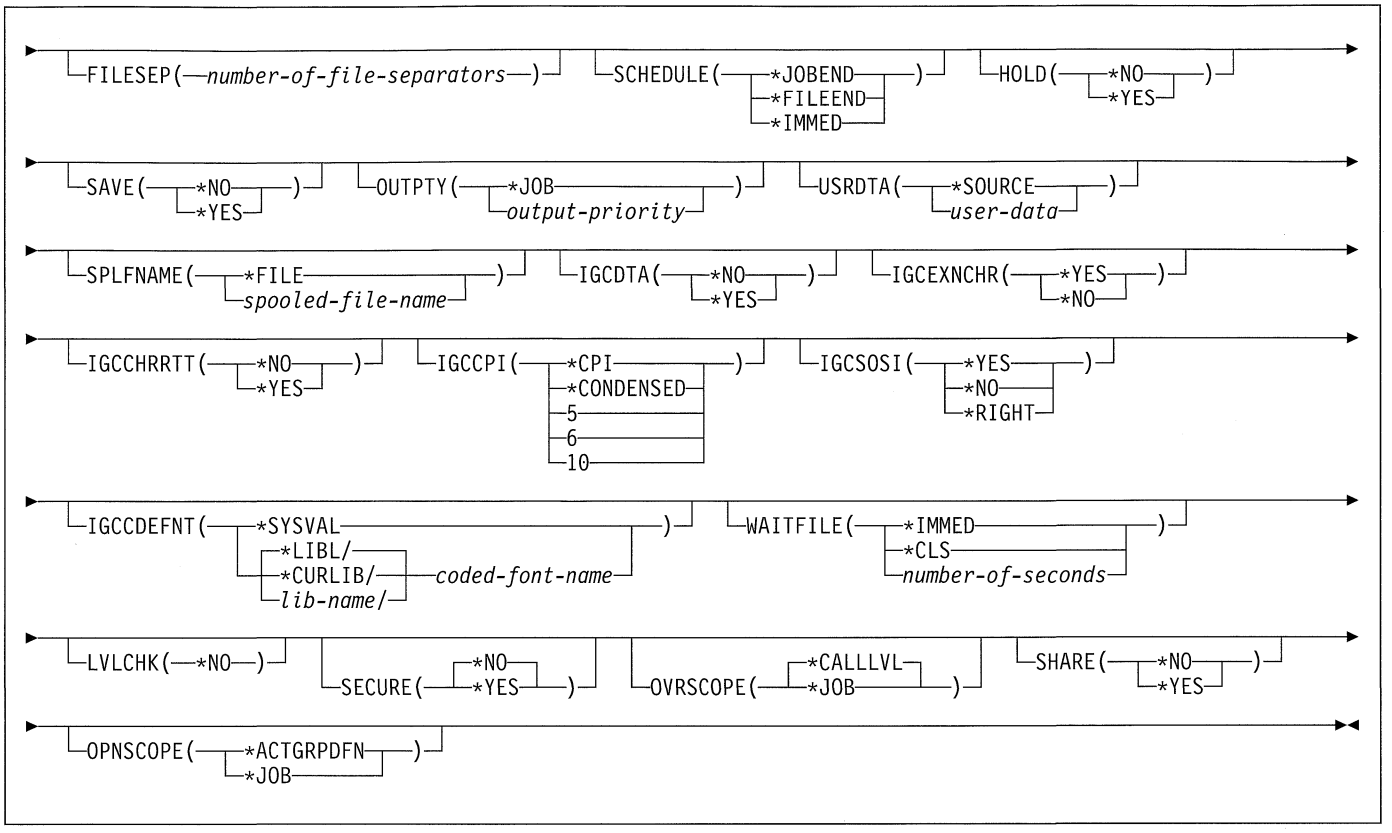
P All parameters preceding this point can be specified in positional form.

1 A maximum of 12 channels with corresponding line numbers are allowed.

2 SCS and IPDS printers only.



Note:
[1] Valid values range from 1 through 255.



Purpose

The Override with Printer File (OVRPRTF) command is used to (1) override (replace) the file named in the program, (2) override certain parameters of a file that are used by the program, or (3) override the file named in the program and override certain parameters of the file processed. Parameters overridden by this command are specified in the file description, in the program, and/or in other file override commands run in the following command.

If a file named in the program is overridden, the name of that file is specified in the FILE parameter and the name of the overriding file (the file processed) is specified in the TOFILE parameter. The OVRPRTF command also specifies parameters to override values contained in the file description of the overriding file. If the file named in the program is not replaced but certain parameters of the file are overridden, the name of the file is specified in the FILE parameter and *FILE is specified in the TOFILE parameter. The parameters overridden are then specified by the other parameters of the OVRPRTF command. Parameters not specified do not affect parameters specified in the file description, in the program, and/or in other file override commands run later.

More information on overriding files is in the *Guide to Programming for Printing*.

Required Parameter

FILE

Specifies the name of the file being used by the program to which this override command is applied. If TOFILE(*FILE) is specified, a display device file must be specified. Otherwise, any device file or database file can be specified.

***PRTF:** The *PRTF file override, which exists in the call level where this command is entered, is applied.

file-override-name: Specify the names of one or more overridden files for which the overrides in the call level are applied.

Optional Parameters

TOFILE

Specifies the qualified name of the printer file that is used instead of the file specified in the FILE parameter. If *FILE is specified, this parameter specifies that certain attributes are overridden by the parameters specified in this command. The parameters specified on this OVRPRTF command override the same parameters specified in the printer file, in the program, and/or in other called OVRPRTF commands.

***FILE:** The printer file named in the FILE parameter has some of its parameters overridden by values specified in this command.

The name of the printer file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

printer-device-file-name: Specify the name of the printer file that is used instead of the overridden file.

DEV

Specifies the name of a printer device description. For nonspooled output, this identifies the printer device used with the printer file to produce the printed output. For spooled output, if OUTQ(*DEVD) is also specified, the default output queue for the specified printer is used for the spooled output data. More information is in the *Guide to Programming for Printing*.

***SYSVAL:** The value specified in the system value QPRTDEV is used.

***JOB:** The printer device specified in the job description is used.

device-name: Specify the name of the printer associated with this display station. The printer and the display station must be attached to the same controller. When printing double-byte character set (DBCS) data, specify a DBCS printer (5553 or 5583).

DEVTYPE

Specifies the type of data stream created for a printer file.

***SCS:** An SNA character stream (SCS) is created. This parameter must be specified when using the 3287, 3812 SCS, 3816 SCS, 4214, 4234 SCS, 4245, 5219, 5224, 5225, 5256, 5262, 6252, or 6262 work station printers.

- If *SCS is specified and the spooled printer file is directed to an IPDS printer, the SCS printer file is converted to emulate an IPDS printer file. More information is in the *Guide to Programming for Printing*.

Double-Byte Character Set Consideration:

When using the 5553 and 5583 DBCS-capable printers, DEVTYPE(*SCS) must be specified.

***IPDS:** An intelligent printer data stream (IPDS) is created. This parameter can be specified when using a 3812 IPDS, 3816 IPDS, 3820, 3825, 3827, 3829, 3831, 3835, 3900, 4028, 4224, 4230, or the 4234 IPDS printer.

- If *IPDS is specified and the spooled printer file is directed to a printer other than an IPDS printer, the IPDS printer file is converted to an SCS printer file. More information is in the *Guide to Programming for Printing*.

***USERASCII:** An ASCII data stream is placed on a spooled output queue. The user is responsible for placing the entire hexadecimal data stream in the buffer, since the AS/400 system does not change or validate the values that are passed. This value cannot be specified with SPOOL(*NO).

***AFPDS:** An advanced function print data stream (AFPDS) is created. This parameter can be specified when using the 3812 IPDS, 3816 IPDS, 3820, 3825, 3827, 3829, 3831, 3835, 3900 4028, 4224, 4230, or the 4234 IPDS printer. The printer must be configured with AFP(*YES).

PAGESIZE

Specifies the length and width of the printer forms used by this printer file. The length is specified in lines per page or by the units specified for the UOM parameter. The width is specified in print positions (characters) per line or by the units specified for the UOM parameter.

The page size must be specified with reference to the way the data is printed on the page. For example, if using 8.5 inch wide by 11.0 inch long forms and printing at 6 lines per inch with a 10-pitch font, specify PAGESIZE(66 85) PAGRTT(0). However, to rotate the page, specify the page size for an 11.0 inch wide by 8.5 inch long page and enter PAGESIZE(51 110) PAGRTT(90).

Note: Specify PAGRTT(*AUTO) or PAGRTT(*DEVD) and PRTQLTY(*DRAFT) on this command to enable automatic reduction or rotation if the data does not fit on the paper.

Specify PAGRTT(*COR) on this command to enable automatic reduction whether or not the data fits on the paper.

This parameter overrides the form size values specified in the printer file, in the program, and/or in other called OVRPRTF commands.

Element 1: Page Length Value

page-length: Specify the page length that is used by this printer file. Although a value ranging from 1 through 255 can be specified as the page length, the value specified must not exceed the actual length of the forms used.

More information about the page lengths that are valid for each printer type is in Appendix B, "Font, Character Identifier, and Other Values Supported for Different Printers."

Element 2: Page Width Value

page-width: Specify the page width used by this printer file. The value specified must not exceed the actual width of the forms used.

More information about page width is in Appendix B, "Font, Character Identifier, and Other Values Supported for Different Printers."

Element 3: Method of Measure

***ROWCOL:** Page length and page width are measured as numbers of rows and columns.

***UOM:** Page length and page width are measured in the units specified on the UOM parameter.

LPI

Specifies the line spacing setting on the printer, in lines per inch, used by this printer file.

The line spacing on the 5256 printer must be set manually. When the lines per inch (LPI) value on this parameter changes (from the value on the previous printer file), an inquiry message is sent to the message queue associated with the printer that requests a change to the LPI value.

The line spacing on the 4214, 4224, 4230, 4234, 4245, and 5262 Printers is set by a print command. These also allow setting the lines per inch spacing on the control panel of the printer. The lines per inch value must not be set at the printer. If the LPI value is overridden at the control panel, the system overrides the value set with the LPI value of the next printer file received.

More information about the lines per page and lines per inch that are valid for each printer type is in Appendix B, "Font, Character Identifier, and Other Values Supported for Different Printers."

| This parameter overrides the overflow value specified in
| the printer file, in the program, and/or in other called
| OVRPRTF commands.

3: The line spacing on the printer is 3 lines per inch. This value is valid only for double-byte character set (DBCS) data.

4: The line spacing on the printer is 4 lines per inch.

6: The line spacing on the printer is 6 lines per inch.

7.5: The line spacing on the printer is 7.5 lines per inch. This value is valid only for double-byte character set (DBCS) printers.

8: The line spacing on the printer is 8 lines per inch.

Note: When printing double-byte character set (DBCS) data for a file specified with LPI(8), use double spacing. Otherwise, the DBCS data does not print correctly. Alphanumeric data, however, prints correctly in single spacing when LPI(8) is specified.

9: The line spacing on the printer is 9 lines per inch.

12: The line spacing on the printer is 12 lines per inch.

CPI

Specifies the printer character density, in characters per inch (CPI), used by this printer file.

For the printers that support fonts, the value specified in the font special value implies the CPI. If FONT(*CPI) is specified, the font used is based on the CPI value. The following diagram describes the default font ID for each CPI value:

CPI	FONT ID DEFAULT
5	245
10	011
12	087
13.3	204
15	222
16.7	400
18	252
20	281

More information about the characters per page and characters per inch that are valid for each printer type is in Appendix B, "Font, Character Identifier, and Other Values Supported for Different Printers."

| This parameter overrides the value specified in the
| printer file, in the program, and/or in other called
| OVRPRTF commands.

5: Double-byte character density is 5 characters per inch.

10: Character density is 10 characters per inch.

12: Character density is 12 characters per inch.

13.3: Character density is 13.3 characters per inch. This value is valid only for double-byte character set (DBCS) printers.

15: Character density is 15 characters per inch.

16.7: Character density is 16.7 characters per inch.

18: Character density is 18 characters per inch. This value is valid only on double-byte character set (DBCS) printers.

20: Character density is 20 characters per inch. This value is valid only on double-byte character set (DBCS) printers.

FRONTMGN

Specifies the offset, down and across, of the origin from the edge on the front side of the paper. The offsets are in the units of measure specified on the UOM parameter. If UOM(*CM) is specified, valid values range from 0 through 57.79, and if UOM(*INCH) is specified, valid values range from 0 through 22.57. This parameter can only be used for printer files with DEVTYPE(*AFPDS) specified.

***DEVD:** The no-print border from the printer is used to place the text on the page when printing to a printer configured as AFP(*YES). A margin of 0 is used for IPDS printers without a no-print border, or which are configured as AFP(*NO).

Element 1: Offset Down

0: No offset of the origin occurs.

offset-down: Specify the offset of the origin from the top of the page.

Element 2: Offset Across

0: No offset of the origin occurs.

offset-across: Specify the offset of the origin from the left side of the page.

BACKMGN

Specifies the offset, down and across, of the origin from the edge on the back side of the paper. The offsets are in the units of measure specified on the UOM parameter. If UOM(*CM) is specified, valid values range from 0 through 57.79, and if UOM(*INCH) is specified, valid values range from 0 through 22.57. This parameter can only be used for printer files with DEVTYPE(*AFPDS) specified.

***FRONTMGN:** The offsets specified on the FRONTMGN parameter are used.

***DEVD:** The no-print border from the printer is used to place the text on the page when printing to a printer configured as AFP(*YES). A margin of 0 is used for IPDS printers without a no-print border, or which are configured as AFP(*NO).

Element 1: Offset Down

0: No offset of the origin occurs.

offset-down: Specify the offset of the origin from the top of the page.

Element 2: Offset Across

0: No offset of the origin occurs.

offset-across: Specify the offset of the origin from the left side of the page.

OVRFLW

Specifies the line number on the current page at which overflow to a new page begins. Generally, after the specified line is printed, the printer overflows to the next page before printing continues. Margins specified for the printer file are ignored when determining overflow. More information is in the *Guide to Programming for Printing*. This parameter overrides the overflow value specified in the printer file, in the program, and/or in other called OVRPRTF commands.

overflow-line-number: Specify the line number on the current page at which overflow to a new page begins, whether or not printing has occurred on that line. The value specified must not be greater than the page length (PAGESIZE). Margins specified for the printer file are ignored when determining overflow.

FOLD

Specifies whether all positions in a record are printed when the record length exceeds the page width (specified by the PAGESIZE parameter). When folding is specified and a record exceeds the page width, any portion of the record that cannot be printed on the first line continues (is folded) on the next line or lines until the entire record has been printed.

The FOLD parameter is ignored under the following conditions:

- When DEVTYPE(*SCS) is not specified.
- When printing through OfficeVision/400.
- When in the S/36 execution environment.

Double-Byte Character Set Considerations:

The system ignores this parameter when printing double-byte character set (DBCS) files. The system assumes that DBCS records fit on a printed line. If the record exceeds the page width, the system continues printing the record on the next line.

This parameter overrides the value specified in the printer file, in the program, and/or in other called OVRPRTF commands.

***YES:** Records whose length exceeds the page width are folded on the following lines. Records whose length exceeds the form width are folded on the following lines.

***NO:** Records are not folded; if a record is longer than the page width, only the part of the record that fits on one line is printed.

RPLUNPRT

Specifies (1) whether unprintable characters are replaced and (2) which substitution character (if any) is used. An *unprintable* character is a character the printer is unable to print.

Double-Byte Character Set Considerations:

For double-byte character set (DBCS) data, an unprintable character is one that cannot be processed. When using DBCS-capable printers, consider the following:

- If IGCEXNCHR(*YES) is also specified, the system replaces unprintable extension characters with DBCS underline characters. There may be some cases in which the system is unable to replace an unprintable character with a DBCS underline character. In this case, the undefined character is printed.
- If IGCEXNCHR(*NO) is also specified, the device replaces all extension characters with the undefined character. Choosing a blank as the replacement character for alphanumeric characters might improve system performance.

More information is in the *Guide to Programming for Printing*.

Element 1: Replace Character

***YES:** Unprintable characters are replaced. The program is not notified when unprintable characters are detected. Note the DBCS considerations above.

***NO:** Unprintable characters are not replaced. When an unprintable character is detected, a message is sent to the program.

Element 2: Replacement Character

' ': Specify, if *YES is also specified on this parameter, that a blank is used as the substitution character when an unprintable character is detected.

'replacement-character': Specify, if *YES is also specified on this parameter, the replacement character that is used each time an unprintable character is detected.

Any printable EBCDIC character can be specified. Valid values range from 40 through 99 and A1 through FE.

ALIGN

Specifies whether the page must be aligned in the printer before printing is started. If ALIGN(*YES) and SPOOL(*NO) are specified, and forms alignment is required, the system sends a message to the message queue specified in the printer device description and waits for a reply to the message. When spool (*YES) is specified on the CRTPRTF command and ALIGN(*FILE) is specified on the STRPRTWTR command, then this parameter is used to determine whether an alignment message is sent by the system.

This parameter is ignored when cut sheets are used (spooled and direct output). Page alignment can be done only for text-only files. Page alignment cannot be done for print jobs containing graphics or bar codes.

| This parameter overrides the alignment value specified
| in the printer file, in the program, and/or in other called
| OVRPRTF commands.

***NO:** No page alignment is required.

***YES:** The page is aligned before the output is printed.

| CTLCHAR

| Specifies whether the printer file supports input with print
| control characters. Any invalid control characters that
| are found are ignored, and single spacing is assumed.

***NONE:** No print control characters are passed in the data being printed.

***FCFC:** The first character of every record contains an American National Standards Institute (ANSI) forms control character. If *FCFC is specified, the record length must include one extra position for the first-character forms-control code. This value is not valid for externally described printer files.

CHLVAL

Specifies a list of channel numbers with their assigned line numbers. Use this parameter only if CTLCHAR(*FCFC) has been specified.

Note: If one or more channel-number/line-number combinations are changed, all other combinations must be re-entered.

***NORMAL:** The default values for skipping to channel identifiers are used. The default values are found in Table 16.

Table 61. ANSI First-Character Forms-Control Codes

Code	Action before Printing a Line
' '	Space one line (blank code)
0	Space two lines
-	Space three lines
+	Suppress space
1	Skip to line 1
2-11	Space one line
12	Skip to overflow line (OVRFLW parameter)

Element 1: Channel Number

channel-number: Specify an American National Standard channel number to be associated with a corresponding 'skip to' line number. Valid values for this parameter range from 1 through 12, corresponding to channels 1 through 12. The CHLVAL parameter associates the channel number with a page line number. For example, if you specify CHLVAL(2 20), channel identifier 2 is allocated with line number 20; therefore, if you place the forms-control 2 in the first position of a record, the printer skips to line 20 before printing the line.

Note: If the printer stops and the next record processed has a channel value forms-control number that is the same value as the line number the printer is on, the printer advances to that value (line number) on the next page. However, if the printer is positioned at the top of the page (line number one) and the channel value forms-control value is associated with line number one, the printer does not advance to a new a new page.

If no line number is specified for a channel identifier, and that channel identifier is encountered in the data, a default of 'space one line' before printing is used. Each channel number can be specified only once.

Element 2: Line Number

line-number: Specify the line number assigned for the channel number in the same list. Valid line numbers range from 1 through 255. If no line number is assigned to a channel number, and that channel number is encountered in the data, a default of 'space one line' before printing is used. Each line number can be specified only once.

FIDELITY

Specifies whether printing continues when print errors are found for printers configured with AFP(*YES).

***CONTENT:** Printing continues when errors are found.

***ABSOLUTE:** Printing stops when errors are found.

FORMFEED

Specifies, for the 4214, 5219, and 5553 printers (including ASCII printers that are configured as an SCS 4214 or SCS 5219 printer), and for IPDS printers, the mode in which forms are fed into the device.

***DEVD:** The forms are fed into the printer in the manner specified in the device description.

***CONT:** Continuous forms are used by the printer. The tractor feed attachment must be on the device.

***CUT:** Single-cut sheets are used by the printer. Each sheet must be manually loaded. For cut sheets, the forms alignment message is not sent.

***AUTOCUT:** The sheet-feed attachment must be on the printer. Single-cut sheets are automatically fed into the printer. The forms alignment message is not sent for cut sheets.

PRTQLTY

Specifies, for the 3812 SCS, 3816 SCS, 4214, 4224, 4230, 4234, and 5219 printers, the quality of print produced.

For the 5219 Printer, different print qualities are produced by varying the speed at which the print ribbon advances. Quality mode (*STD or *NLQ) results in normal print ribbon advancement. In draft mode (*DRAFT), the ribbon advances at a rate of one-third the distance it advances in quality mode. The 5219 Printer has a conserve ribbon switch that overrides the value of *DRAFT specified by this parameter.

For the 3812 SCS and 3816 SCS Printers, the automatic hardware selection of computer output reduction printing selected through soft switches on the printers occurs only when *DRAFT is specified for PRTQLTY and PAGRTT is *DEVD. If PAGRTT(*COR) is specified, the PRTQLTY parameter does not affect the printed output.

For the 4214, 4224, 4230, and 4234 Printers, standard print quality is produced by varying the density of the dot matrix pattern used to create printable characters. Standard mode (*STD) is the normal mode. Quality mode (*NLQ) requires multiple passes by the printer to produce a line of data. Draft mode (*DRAFT) results in high-speed printing.

More information about the valid values for the 4214, 4224, 4230, 4234, and 5219 Printers is in Appendix B, "Font, Character Identifier, and Other Values Supported for Different Printers."

Notes:

1. For the 4214 Printer, standard print quality (*STD or *NLQ) is only supported for 10 and 12 characters per inch. If PRTQLTY(*STD or *NLQ) and 5, 15, or 16.7 characters per inch is specified, the data is printed in draft mode.
2. For the 4234 Printer, only a limited character set (62 characters) is supported when PRTQLTY(*DRAFT) is specified. A description of the character set supported with draft print quality is in the *4234 Printer Operator's Guide.*
3. For the 4224 and 4230 printers, the fonts supported are not available for all three print qualities. The OCR-A and OCR-B fonts are supported only with PRTQLTY(*NLQ). The Courier and Essay fonts are available only with PRTQLTY(*NLQ) and PRTQLTY(*STD). The Gothic font is available only with PRTQLTY(*DRAFT) or PRTQLTY(*FASTDRAFT). If there is a mismatch between the print quality and the font selected, the font is changed to match the print quality.
4. Specify PAGRTT(*DEVD) and PRTQLTY(*DRAFT) on this command to enable automatic rotation if the data does not fit on the paper.

***STD:** The output is printed with standard quality.

***DRAFT:** The output is printed with draft quality.

***DEVD:** The print quality is set on the printer by the user, if it is not set within the data stream.

***NLQ:** The output is printed with near letter quality.

***FASTDRAFT:** The output is printed at a higher speed and with lower quality than it would be if you specified *DRAFT. This value is only supported by the 4230 printer.

DRAWER

Specifies the source drawer used when automatic cut sheet feed mode is used (specified by FORMFEED(*AUTOCUT)).

1: The paper is fed from the first drawer on the sheet-feed paper handler.

2: The paper is fed from the second drawer on the sheet-feed paper handler.

3: The paper is fed from the third drawer on the sheet-feed paper handler.

***E1:** The envelopes are fed from the envelope drawer on the sheet-feed paper handler.

FONT

Specifies the font identifier and point size used with this printer device file. If a font identifier or point size is not specified, the system automatically sets them.

More information about the valid font identifiers, the display value, the characters per inch value implied with each font style, a description of each font style, and whether the font is supported on a particular printer is in Appendix B, "Font, Character Identifier, and Other Values Supported for Different Printers."

Note: Some fonts can be substituted by the printer. Consult the various printer reference guides for details.

***CPI:** The identifier of the font with the specified pitch (characters per inch (CPI)) is used.

***DEVD:** The font identifier and point size specified in the device description are used.

Element 1: Font Identifier

identifier: Specify the numeric font identifier associated with this printer.

Element 2: Point Size

***NONE:** The point size is supplied by the system and is determined by the specified font identifier.

point-size: Specify a point size ranging from 0.1 through 999.9.

CHRID

Specifies the character identifier (graphic character set and code page) for the file. This parameter allows printing of text that is in different character identifier (graphic character set and code page) coding. The value specified on this parameter is used to instruct the printer device to interpret the hexadecimal byte string to print the same characters that were intended when the text was created. More information about the character identifier is in the *Guide to Programming for Printing*. A list of valid CHRID values and applicable printers is in the "CHRID Values and Applicable Printers (CHRID parameter)" table in Appendix B, "Font, Character Identifier, and Other Values Supported for Different Printers."

***DEVD:** The default CHRID value that the device is designed to handle is used. The *DEVD value means character selection is normal because the file has the same character identifier as the device default.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***JOBCCSID:** The character identifier for the printer file is taken from the coded character set identifier (CCSID) of the job.

Note: This value is not allowed if the file was created on a system at an earlier release level than V2R3M0.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values that match the attributes of the printer. Valid values range from 1 through 32767.

Element 2: Code Page

code-page: Specify the code page value that matches the attributes of the printer. Valid values range from 1 through 32767.

FNTCHRSET

Specifies a downloaded font consisting of a character set and code page. This parameter can only be used for printer files with DEVTYPE(*AFPDS) specified.

***FONT:** The value specified on the FONT parameter is used.

Element 1: Font Character Set

The name of the font character set can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

character-set: Specify the font character set to use.

Element 2: Code Page Name

The name of the code page name can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

code-page: Specify the code page value used to create the command parameters. Valid values range from 1 through 999.

CDEFNT

Specifies the coded font that the system uses for single-byte character set (SBCS) printing. This parameter can only be used for printer files with DEVTYPE(*AFPDS) specified.

***FNTCHRSET:** The font specified on the FNTCHRSET parameter is used.

The name of the coded font name can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

coded-font-name: Specify the DBCS-coded font name to use.

PAGRTT

Specifies the degree of text rotation for the 3812, 3816, 4028, 3820, 3825, 3827, 3829, 3831, 3835, and 3900 printers. This parameter allows the user to specify the degree of rotation of the text on the page with respect to the way the form is loaded into the printer. See the note under the PAGESIZE parameter for directions on specifying page size when rotating the page.

Specify *AUTO or *DEVD for this parameter and PRTQLTY(*DRAFT) on this command to enable automatic rotation if the data does not fit on the paper.

***AUTO:** Indicates that automatic rotation of output is done to fit the printed data on the form. If rotation does not accomplish this, computer output reduction is per-

formed automatically (regardless of the print quality being used). This parameter is valid only for printers supporting rotation.

***DEVD:** The operating system sends a device default rotation value to the printer. Page rotation is dependent upon your printer's specifications. See your printer or printer emulation documentation to determine how page rotation is affected.

***COR:** Computer output reduction is done. Computer output reduction allows printed output intended for a 13.2 inch wide by 11.0 inch long form to be printed on an 8.5 inch wide by 11.0 inch long form.

For computer output reduction printing, the following operations are done by the 3812, 3816, 4028, 3820, 3825, 3827, 3829, 3831, 3835, and 3900 printers:

- Automatic rotation to *COR is not done if the file contains graphics, bar codes, variable LPI, variable font, variable page rotations, or variable drawer.
- The text is rotated 90 degrees clockwise from the 0 degree rotation position (lower left corner of the first edge loaded into the printer).

  Note: For landscape paper on a 3835 printer, the rotation is counter-clockwise from the 0 degree rotation position (upper right corner of the first edge loaded into the printer).
- A top and left margin of 0.5 inches is added to the printed output.
- The 12-pitch fonts are changed to a 15-pitch font and 15-pitch fonts are changed to a 20-pitch font. All other font widths are changed to a 13.3-pitch font, except for the 4028 printer where they are changed to a 15-pitch font.
- Vertical spacing (specified by the LPI parameter) is 70 percent of the normal spacing.
- The page size is set to 8.5 inches wide by 11 inches long.

0: No rotation occurs. Printing starts at and is parallel to the edge loaded into the printer.

90: Rotation of the text is done 90 degrees clockwise from the 0 degree writing position.

180: Rotation of the text is done 180 degrees clockwise from the 0 degree writing position.

270: Rotation of the text is done 270 degrees clockwise from the 0 degree writing position.

MULTIUP

Specifies, for spooled output only, the number of pages printed on a single physical page.

Note: Overlays are not reduced when more than one page is printed on a side.

1: One page of output is printed on one physical sheet of paper.

2: Two pages of output are printed on 1 physical sheet of paper.

4: Four pages of output are printed on 1 physical sheet of paper.

PRTTXT

Specifies up to 30 characters of text to be printed at the bottom of each page of output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***JOB:** The value for the current job is used.

***BLANK:** Text is not specified.

'print-text': Specify the character string printed at the bottom of each page. No more than 30 characters of text can be entered, enclosed in apostrophes.

JUSTIFY

Specifies the printing positions of the characters on a page so the right-hand margin of printing is regular. Justification is done to the record length on the printer file opened.

Note: The JUSTIFY parameter is supported only on the 3812 SCS, 3816 SCS, and 5219 Printers.

0: No justification occurs.

50: Spaces are added to the blanks in the text so that the right margin is more closely aligned but not flush.

100: The text is expanded by spaces (added where the blanks already exist) until the right margin is flush.

DUPLEX

Specifies whether output is printed on one side or two sides of the paper.

***NO:** The output is printed on one side of the paper.

***YES:** The output is printed on both sides of the paper with the top of each printed page at the same end of the paper.

***TUMBLE:** The output is printed on both sides of the paper with the top of one printed page at the opposite end of the sheet from the top of the other printed page. This is usually used for output that is bound at the top.

DFRWRT

Specifies how much output is held in the system buffer before being sent to the printer.

Note: If this command is specified, the Display Override (DSPOVR) command will either display or print the override along with any others that are specified on this command.

***YES:** The system controls the amount of output that is held in the buffer before being sent to the printer.

If SPOOL(*YES) is specified along with SCHEDULE(*IMMED), output is held in the buffer until a page of output is available or until the system buffer is full.

***NO:** If SPOOL(*NO) is specified, output is not held in the buffer. Output is sent to the printer immediately after the program performs a write operation.

If the spooled output schedule is not immediate, specifying DFRWRT(*NO) has no effect.

SPOOL

Specifies whether the output data for the printer file is spooled. If SPOOL(*NO) is specified, the following parameters in this command are ignored: OUTQ, COPIES, MAXRCDS, FILESEP, SCHEDULE, HOLD, SAVE, OUTPTY, and USRDTA.

Note: If SPOOL(*NO) is the current value in the printer file, or if SPOOL(*NO) is specified in this or any other OVRPRTF command that is in effect when the file is opened, the following parameters that apply to spooled files are ignored: OUTQ, COPIES, MAXRCDS, FILESEP, SCHEDULE, HOLD, SAVE, OUTPTY, USRDTA, and SPLFNAME. This parameter overrides the spool value specified in the printer file, and/or in other called OVRPRTF commands.

***YES:** The data is spooled for processing by a diskette writer or a print writer.

***NO:** The data is not spooled; it is sent directly to the device and printed as the output becomes available.

OUTQ

Specifies, for spooled output only, the qualified name of the output queue. This parameter overrides the output queue name specified in the printer file and/or in other called OVRPRTF commands.

***DEV:** The output queue specified on the PRTDEV parameter is used.

***JOB:** The output queue associated with the job is used.

The name of the output queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

output-queue-name: Specify the name of the output queue to which the output data is spooled.

FORMTYPE

Specifies the type of form on which the output is printed. The identifiers used to indicate the type of forms are user-defined and can be a maximum of 10 characters in length.

Note: If a form type other than *STD is specified, the system (when the output is produced) sends a message that identifies the form type to the system operator, and requests that the specified type of forms be put in the printer. This parameter overrides the form type value specified in the printer file and/or in other called OVRPRTF commands.

***STD:** The standard form type is used. The output is printed on the form type specified in the printer file for the printers selected. The printer file contains information that controls how the document is printed on a particular printer.

form-type: Specify the identifier of the form type used with this device file for printed output from jobs. Up to 10 alphanumeric characters can be specified. When the device file is opened, the system sends a message identifying the form type to the system operator, and requests that the identified forms be in the printer.

COPIES

Specifies, for spooled files, the number of copies being printed.

Note: This parameter overrides the copy value specified in the printer file.

number-of-copies: Specify a value, ranging from 1 through 255, that indicates the number of identical printouts produced when this printer file is used.

PAGERANGE

Specifies the page range to print for each copy of the file to be printed.

Element 1: Starting Page to Print

***ENDPAGE:** Only the ending page is printed.

starting-page: Specify the page on which to start printing.

Element 2: Ending Page to Print

***END:** The last page in the file is printed.

ending-page: Only the ending page is printed.

FRONTOVL

Specifies the qualified name of the object that contains both the overlay that is printed on the FRONT side of the page and the offset, down and across, from the point of origin used when the overlay is printed.

***NONE:** No overlay is used.

Element 1: Overlay Name

The name of the overlay can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

overlay-name: Specify the name of the overlay.

Element 2: Offset Down

0: No offset down from the point of origin is used.

offset-down: Specify the offset down from the point of origin at which to begin printing the overlay. If UOM(*CM) is specified, valid values range from 0 through 57.79, and if UOM(*INCH) is specified, valid values range from 0 through 22.57.

Element 3: Offset Across

0: No offset across from the point of origin is used.

offset-across: Specify the offset across from the point of origin at which to begin printing the overlay. If UOM(*CM) is specified, valid values range from 0 through 57.79, and if UOM(*INCH) is specified, valid values range from 0 through 22.57.

BACKOVL

Specifies the object name and library name containing both the overlay that is printed on the BACK side of the page and the offset, down and across, from the point of origin used when the overlay is printed.

***FRONTOVL:** The values that are specified on the FRONTOVL parameter are used.

***NONE:** No overlay is used.

Element 1: Overlay Name

The name of the overlay can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

overlay-name: Specify the name of the overlay.

Element 2: Offset Down

0: No offset down from the point of origin is used.

offset-down: Specify the offset down from the point of origin at which to begin printing the overlay. If UOM(*CM) is specified, valid values range from 0 through 57.79, and if UOM(*INCH) is specified, valid values range from 0 through 22.57.

Element 3: Offset Across

0: No offset across from the point of origin is used.

offset-across: Specify the offset across from the point of origin at which to begin printing the overlay. If UOM(*CM) is specified, valid values range from 0 through 57.79, and if UOM(*INCH) is specified, valid values range from 0 through 22.57.

UOM

Specifies the unit of measure that is used.

***INCH:** An inch is used as the unit of measure.

***CM:** A centimeter is used as the unit of measure.

MAXRCDS

Specifies, for spooled output only, the maximum number of records that can be in the spooled file for jobs using this printer file. If this maximum is reached, an inquiry message is sent to the program message queue. This parameter overrides the value specified in the printer file and/or in other called OVRPRTF commands.

***NOMAX:** The system maximum is used.

maximum-records: Specify a value, ranging from 1 through 999999, that specifies the maximum number of records allowed in the spooled file.

FILESEP

Specifies, for spooled output only, the number of separator pages placed at the start of each printed file, including those between multiple copies of the same output. Each separator page has the following items printed on it: file name, file number, job name, user name, and the job number. This parameter overrides the separator value specified in the printer file and/or in other called OVRPRTF commands.

number-of-file-separators: Specify the number of separator pages used at the start of each printed output file produced by this device file. Valid values range from 0 through 9. If 0 is specified, no separator pages are printed for the file. In this case, the printed output for each file (or copy of a file) starts at the top of a new page.

SCHEDULE

Specifies, for spooled output only, when the spooled file is available to a writer. This parameter overrides the scheduling value specified in the printer file and/or in other called OVRPRTF commands.

***JOBEND:** The spooled file is made available to the writer only after the entire job is completed.

***FILEEND:** The spooled file is made available to the writer as soon as the file is closed in the program.

***IMMED:** The spooled file is made available to the writer as soon as the file is opened in the program.

HOLD

Specifies, for spooled output only, whether the spooled file is held. The spooled file can be released by using the Release Spooled File (RLSSPLF) command.

Note: This parameter overrides the hold value specified in the printer file and/or in other called OVRPRTF commands.

***NO:** The spooled printer file is not held by the output queue. The spooled output is available to a writer based on the SCHEDULE parameter value.

***YES:** The spooled file is held until released by the Release Spool File (RLSSPLF) command.

SAVE
Specifies, for spooled output only, whether the spooled file is saved (left on the output queue) after the output has been produced. This parameter overrides the save value specified in the printer file and/or in other called OVRPRTF commands.

***NO:** The spooled file data is not saved on the output queue after it has been produced.

***YES:** The spooled file data is saved on the output queue until the file is deleted. After the file is produced, the number of copies (see COPIES parameter) is set to 1, and its status is changed from WTR to SAV. Refer to the Release Spooled File (RLSSPLF) command for information on how to produce the spooled file again.

OUTPTY
Specifies the output priority for spooled output files that are produced by this job. The highest priority is 1 and the lowest priority is 9. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***JOB:** The output priority associated with the job that created the spooled file is used.

output-priority: Specify the output priority. Valid values range from 1 (high priority) through 9 (low priority).

USRDTA
Specifies, for spooled output only, the user-specified data that identifies the file.

***SOURCE:** If the spooled file was created by an application program, the name of the program is used. Otherwise, blanks are used.

user-data: Specify up to 10 characters of text.

SPLFNAME
Specifies, for spooled output only, the spooled file name.

***FILE:** The name of the printer file is used for the spooled file name.

spooled-file-name: Specify up to 10 characters for the spooled file name.

IGCDTA
Specifies, for program-described original files, whether the file processes double-byte character set (DBCS) data. For externally described printer files, this parameter specifies DBCS attributes of the file.

For program-described files:

***NO:** The file does not process DBCS data.

***YES:** The file processes DBCS data.

IGCEXNCHR
Specifies whether the system processes double-byte character set (DBCS) extension characters.

***YES:** The system processes DBCS extension characters.

***NO:** The system does not process DBCS extension characters; it prints extension characters as the undefined character.

IGCCHRRTT
Specifies, for the 5553 and 5583 Printers only, whether the printer rotates double-byte characters 90 degrees counterclockwise when printing. The system prints rotated double-byte characters so they appear in a vertical reading sequence. Alphanumeric characters are not rotated.

***NO:** The system does not rotate double-byte characters when printing.

***YES:** The system rotates double-byte characters 90 degrees counterclockwise when printing. The printer rotates each character individually.

IGCCPI
Specifies the printer character density of double-byte character set (DBCS) characters, in characters per inch (CPI).

Note: This parameter does not specify the printer character density of alphanumeric characters. Alphanumeric characters are printed with the value specified on the CPI parameter.

***CPI:** DBCS character density is based on the values specified for the CPI parameter. The system prints one double-byte character for every two alphanumeric characters.

- For CPI(10), DBCS characters print at 5 characters per inch.
- For CPI(12), DBCS characters print at 6 characters per inch.
- For CPI(13.3), DBCS characters print at 6.7 characters per inch (same as IGCCPI(*CONDENSED)).
- For CPI(15), DBCS characters print at 7.5 characters per inch.
- For CPI(18), DBCS characters print at 9 characters per inch.
- For CPI(20), DBCS characters print at 10 characters per inch.

***CONDENSED:** Condensed printing is used in which the system prints 20 DBCS characters every 3 inches. This value is valid only for the 5553 or 5583 Printers.

5: Double-byte character density is 5 characters per inch.

6: DBCS character density is 6 characters per inch. This value is valid for the 5553 and 5583 Printers only.

10: DBCS character density is 10 characters per inch. This value is valid for the 5553 or 5583 Printers only.

IGCSOSI
Specifies, for bracketed DBCS character strings only, how the system prints shift control characters.

***NO:** The system does not print shift control characters. These characters do not occupy a position in printed output.

***YES:** The system prints shift control characters as blanks.

***RIGHT:** The system prints two blanks when printing shift-in characters but does not print shift-out characters.

IGCCDEFNT

Specifies the coded font that the system uses for DBCS printing.

***SYSVAL:** The DBCS-coded font specified in the system value QIGCCDEFNT is used.

The name of the coded font name can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

coded-font-name: Specify the coded font name to use.

WAITFILE

Specifies the number of seconds that the program waits for the file resources and session resources to be allocated when the file is opened, or for the device or session resources to be allocated when an acquire operation is performed to the file. If those resources are not allocated within the specified wait time, an error message is sent to the program. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: An immediate allocation of the device by the device resource is required when an acquire operation is performed to the file.

This parameter overrides the wait time specified in the printer file, in the program, and/or in other called OVRPRTF commands.

***IMMED:** The program does not wait; when the file is opened, an immediate allocation of the file resources is required.

***CLS:** The job default wait time is used as the wait time for the file resources being allocated.

number-of-seconds: Specify the number of seconds that the program waits for the file resources to be allocated to the printer file when the file is opened, or the wait time for the device allocated when an acquire operation is performed to the file. Valid values range from 1 through 32767 seconds.

LVLCHK

Specifies whether the record format level identifiers in the program are checked against those in the device file when the file is opened. If so, the record format identifiers in the program must match those in the device file. Because the same record format name can exist in more than one file, each record format is given an internal system identifier when it is created.

Note: This parameter overrides the value specified in the printer file, in the program, and/or in other called OVRPRTF commands. Level checking cannot be done unless the program contains the record format identifiers. This command cannot override level checking from *NO to *YES.

***NO:** The level identifiers are not checked when the file is opened.

SECURE

Specifies whether this file is safe from the effects of previously called file override commands. If SECURE is not specified, processing occurs as if SECURE(*NO) is specified.

***NO:** This file is not protected from the effects of other file overrides; its values can be overridden by the effects of previously called file override commands.

***YES:** This file is protected from the effects of any file override commands previously called.

OVRSCOPE

Specifies the extent of influence (scope) of the override.

***CALLLVL:** The scope of the override is determined by the current call level. All open operations done at a call level that is the same as or higher than the current call level are influenced by this override.

***JOB:** The scope of the override is the job in which the override occurs.

SHARE

Specifies whether the open data path (ODP) for the printer file is shared with other programs in the routing step. When an ODP is shared, the programs accessing the file share facilities such as the file status and the buffer.

More information on shared database files is in the *Database Guide*.

***NO:** The ODP created by the program with this attribute is not shared with other programs in the routing step. Every time a program opens the file with this attribute, a new ODP to the file is created and activated.

Note: This includes several opens in the same program.

***YES:** The ODP created with this attribute is shared with each program in the routing step that also specifies SHARE(*YES) when it opens the file.

Note: When SHARE(*YES) is specified and control is passed to a program, a read operation in that program retrieves the next input record. A write operation produces the next output record.

OPNSCOPE
Specifies the extent of influence (scope) of the open operation.

***ACTGRPDFN:** The scope of the open operation is determined by the activation group of the program that called the OVRPRTF command processing program. If the activation group is the default activation group, the scope is the call level of the caller. If the activation group is a non-default activation group, the scope is the activation group of the caller.

***JOB:** The scope of the open operation is the job in which the open operation occurs.

Examples

Example 1: Printing Output

```
OVRPRTF  FILE(PRINTOUT)  TOFILE(PRINT3)  SPOOL(*YES)
  COPIES(5)  OUTQ(OUTPUT1)
```

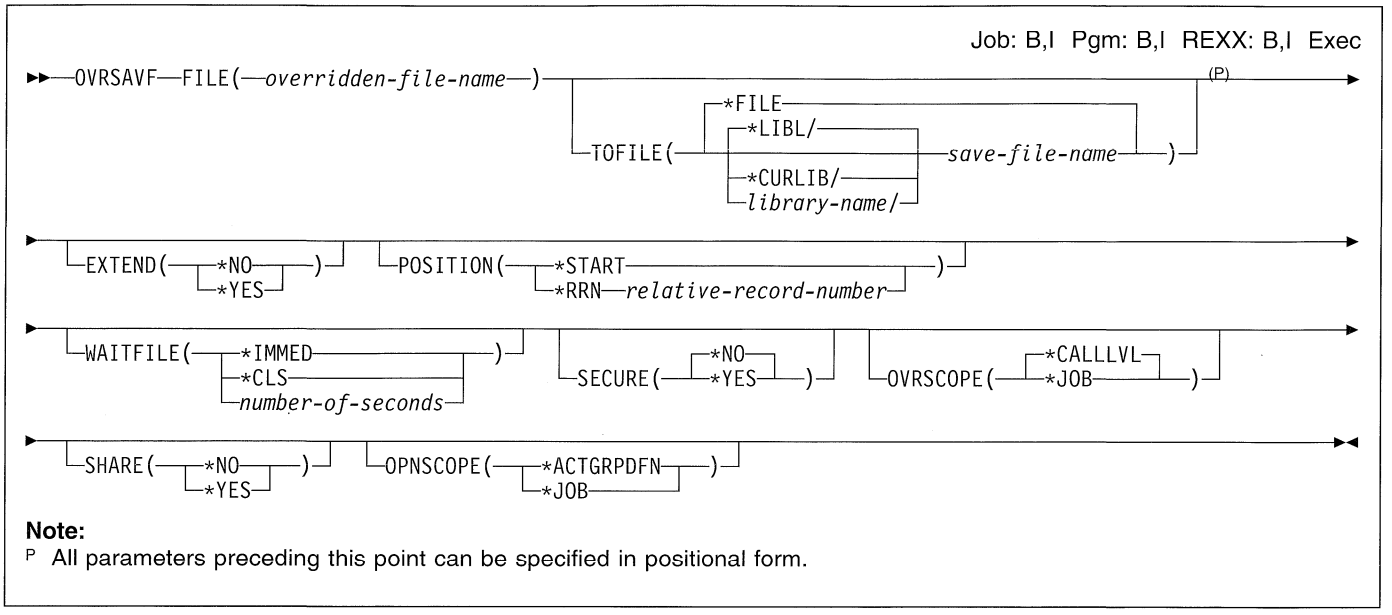
This command overrides the file named PRINTOUT and uses the printer file named PRINT3 to produce the spooled output on the printer. The output from the program is sent to the OUTPUT1 output queue. Five copies of the spooled file are printed on the printer specified on the Start Printer Writer (STRPRTWTR) command.

Example 2: Rotating Double-Byte Characters

```
OVRPRTF  FILE(IGCLIB/IGCPRT)  IGCDTA(*YES)
  IGCCHRRTT(*YES)
```

This command overrides the IGCPRT printer file, which is stored in the IGCLIB library. The override puts the IGCALTTYP DDS keyword into effect to change character output fields to DBCS fields, and rotates the double-byte characters when printing.

OVRSAVF (Override with Save File) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Override with Save File (OVRSAVF) command is used to (1) override or replace a file named in a program, (2) override certain attributes of a file that are used by a program, or (3) override the file named in a program and certain attributes of the overriding file.

Note: Overrides do not apply to save and restore commands.

More information on overriding files is in the *Data Management Guide*.

Required Parameter

FILE

Specifies the name of the file being used by the program to which this override command is applied. If TOFILE(*FILE) is specified, a display device file must be specified. Otherwise, any device file or database file can be specified.

Note: The information in a save file has meaning only to Operating System/400 save and restore; redirecting another type of file to a save file or vice versa is not recommended.

Optional Parameters

TOFILE

Specifies the name of the save file that is used instead of the file specified in the FILE parameter or, if *FILE is specified, specifies that certain attributes are overridden by parameters specified on this command. The parameters specified on this command override the other values specified in the save file or in the program.

***FILE:** The save file named in the FILE parameter has certain parameters overridden by the values specified in this command.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the qualified name of the save file that is used instead of the overridden file name. If no library qualifier is given, *LIBL is used to find the save file.

EXTEND

Specifies, for output operations only, whether new records are added to the end of the data currently in the save file. This option is used to start processing after an application or system failure. When this operation is completed, the file must contain the image of a single save operation made by a save command, or it may not be possible to restore objects from the save file. This parameter overrides the extend value specified in the program. The sequencing information in the file's records guarantees that after a system failure, a record cannot be skipped or sent twice.

***NO:** Records are not added to the end of the specified save file, but they replace existing records in the file. If

the save file already contains records, an inquiry message is sent that clears the file or cancels the operation. If no EXTEND value is specified by the program or in an override, this is the default action assumed when the file is opened for output.

***YES:** New records are added to the end of the records already contained in the save file.

POSITION

Specifies the starting position for getting records from the save file. The first record to get is either at the beginning of the file (*START) or at a particular relative record number position in the file (*RRN). This parameter overrides the value specified in the program.

***START:** Get the first record in the file first. If no POSITION value is specified by the program, or in an override, this is the default action assumed when the file is opened for input.

***RRN** *relative-record-number:* Specify the record number (its position from the beginning of the file) of the record that the user gets first. The value *RRN must go before the relative record number. For example, POSITION(*RRN 480) specifies that the 480th record in the file is gotten first.

WAITFILE

Specifies the number of seconds that the program waits for the file resources and session resources to be allocated when the file is opened, or for the device or session resources to be allocated when an acquire operation is performed to the file. If those resources are not allocated within the specified wait time, an error message is sent to the program. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: An immediate allocation of the device by the device resource is required when an acquire operation is performed to the file.

This parameter overrides the wait time specified in the program or in the save file.

***IMMED:** The program does not wait; when the file is opened, an immediate allocation of the file resources is required.

***CLS:** The job default wait time is used as the wait time for the file resources being allocated.

number-of-seconds: Specify the number of seconds that the program waits for the file resources to be allocated. Valid values range from 1 through 32767 seconds.

SECURE

Specifies whether this file is safe from the effects of previously called file override commands. If SECURE is not specified, processing occurs as if SECURE(*NO) is specified.

***NO:** This file is not protected from the effects of other file overrides; its values can be overridden by the effects of previously called file override commands.

***YES:** This file is protected from the effects of any file override commands previously called.

OVRSCOPE

Specifies the extent of influence (scope) of the override.

***CALLLVL:** The scope of the override is determined by the current call level. All open operations done at a call level that is the same as or higher than the current call level are influenced by this override.

***JOB:** The scope of the override is the job in which the override occurs.

SHARE

Specifies whether the open data path (ODP) for the save file is shared with other programs in the routing step. When an ODP is shared, the programs accessing the file share facilities such as the file status and the buffer.

More information on shared database files is in the *Database Guide*.

***NO:** The ODP created by the program with this attribute is not shared with other programs in the routing step. Every time a program opens the file with this attribute, a new ODP to the file is created and activated.

Note: This includes more than one open in the same program.

***YES:** The ODP created with this attribute is shared with each program in the routing step that also specifies SHARE(*YES) when it opens the file.

Note: When SHARE(*YES) is specified and control is passed to a program, a read operation in that program retrieves the next input record. A write operation produces the next output record.

OPNSCOPE

Specifies the extent of influence (scope) of the open operation.

***ACTGRPDFN:** The scope of the open operation is determined by the activation group of the program that called the OVRSAVF command processing program. If the activation group is the default activation group, the scope is the call level of the caller. If the activation group is a non-default activation group, the scope is the activation group of the caller.

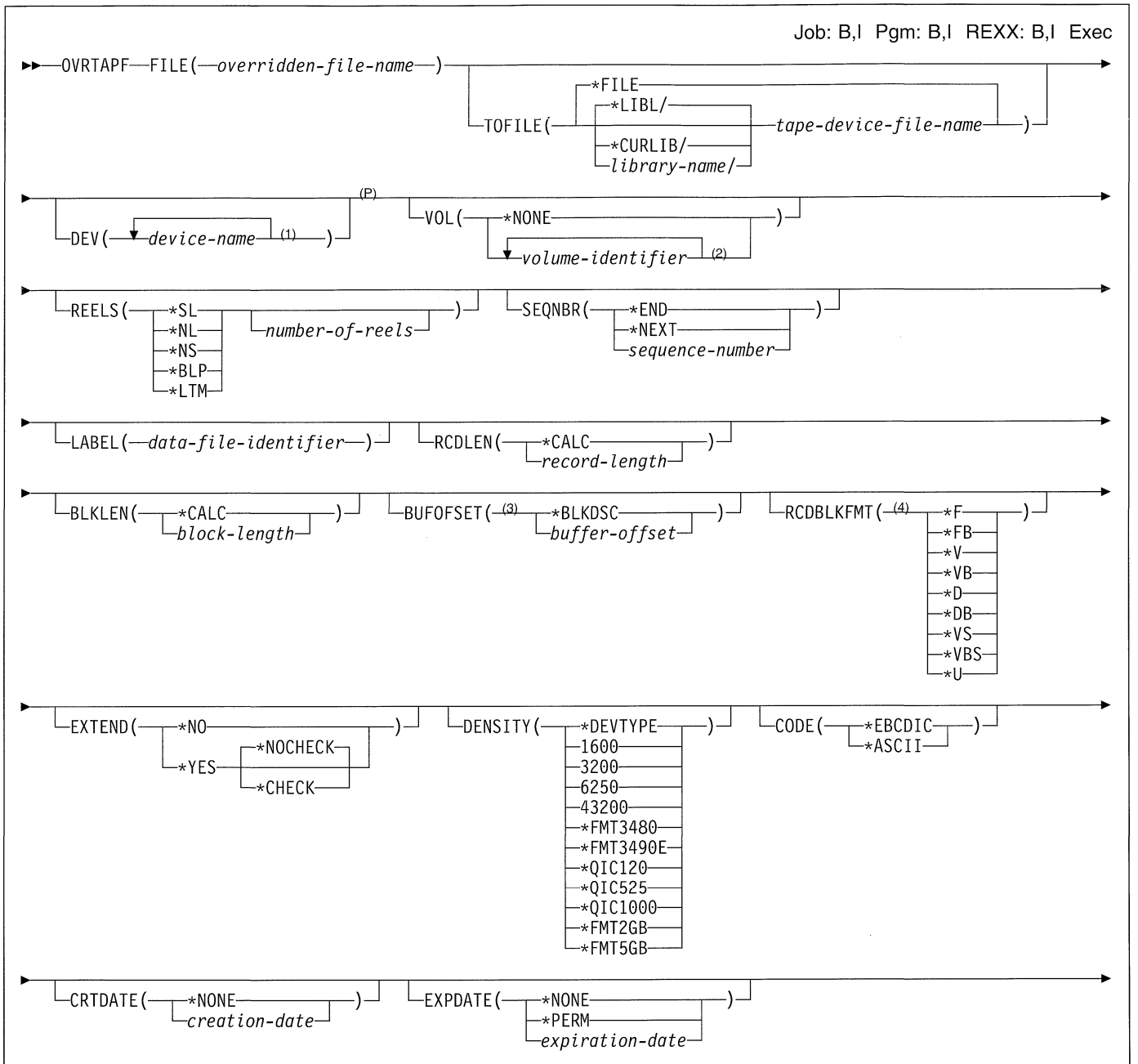
***JOB:** The scope of the open operation is the job in which the open operation occurs.

Example

```
OVRSAVF   FILE(ONLINE)  POSITION(*RRN 100)  SECURE(*YES)
```

This command overrides the file named ONLINE so that the first record gotten after the file is opened for input is relative record number 100. The file is also safe from overrides (in previous program calls).

OVRTAPF (Override with Tape File) Command



Notes:

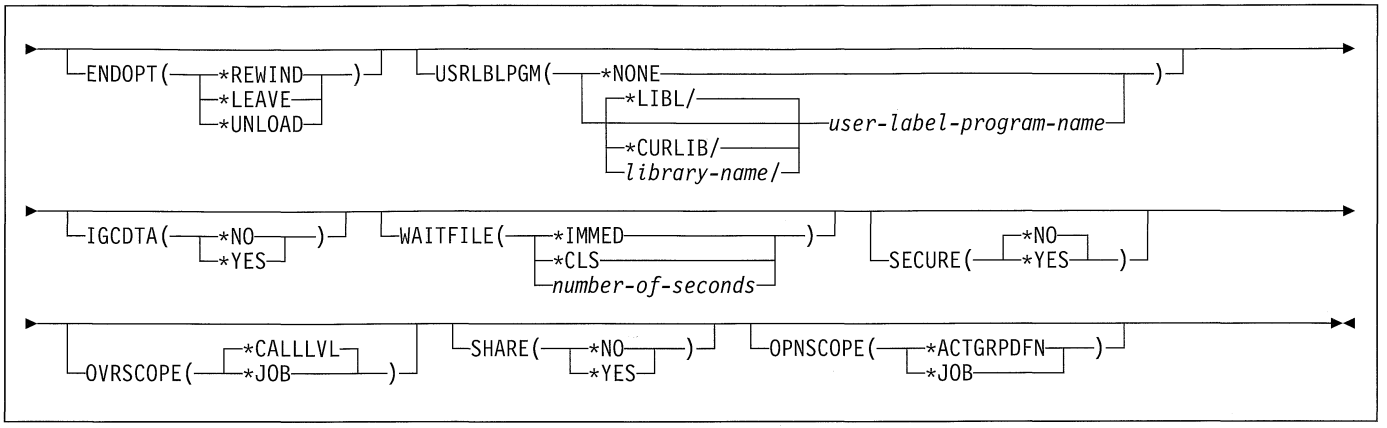
1 A maximum of 4 repetitions

P All parameters preceding this point can be specified in positional form.

2 A maximum of 50 repetitions

3 The value *BLKDSC is valid only for a file with record block format *D or *DB.

4 The values *F, *FB, *VS, *VBS, and *U are valid for both EBCDIC and ASCII codes; *V and *VB are valid only for EBCDIC; *D and *DB are valid only for ASCII.



Purpose

The Override with Tape File (OVRTAPF) command is used to (1) override/replace a file named in a program, (2) override certain attributes of a file that is used by a program, or (3) override the file named in a program and override certain attributes of the file processed.

Parameters overridden by this command are specified in the file description, in the program, and/or in other called file override commands. If a file named in the program is overridden, the name of that file is specified in the FILE parameter and the name of the overriding file is specified in the TOFILE parameter. The OVRTAPF command can also specify parameters to override values contained in the file description of the overriding file. If the file named in the program is not replaced, but certain parameters of the file are overridden, the name of the file is specified in the FILE parameter and *FILE is specified in the TOFILE parameter. The parameters overridden are then specified by the other parameters of the OVRTAPF command. Parameters that are not specified do not affect the parameters specified in the file description, in the program, and/or in other called file override commands.

More information on overriding files is in the *Data Management Guide*, the *Guide to Programming Displays*, and the *Guide to Programming for Printing*.

Restriction: Non-labeled tapes cannot be duplicated to 1/4 inch or 8mm cartridge devices.

Required Parameter

FILE

Specifies the name of the file being used by the program to which this override command is applied. If TOFILE(*FILE) is specified, a display device file must be specified. Otherwise, any device file or database file can be specified.

Optional Parameters

TOFILE

Specifies the qualified name of the tape device file that is used instead of the file specified in the FILE parameter, or if *FILE is specified, specifies that certain attributes are overridden by parameters specified in this command. The parameters specified on this OVRTAPF command override the other values specified in the tape device file or in the program.

***FILE:** The tape device file named in the FILE parameter has some of its parameters overridden by values specified in this command.

The name of the tape device file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

tape-device-file-name: Specify the qualified name of the Tape device file that is used instead of the overridden file.

DEV

Specifies the names of one or more tape devices used with this tape device file to perform input/output data operations. The device names in the OVRTAPF command (up to four) override the device names specified in the program or in the tape device file.

device-name: Specify the names of one or more devices (no more than four) used with this tape device file. The order in which the device names are specified is the order in which tapes on the devices are processed. When more volumes are processed than the number of devices in the DEV list, the devices are used in the same order specified, wrapping around to the first device as needed. Each device name must be known on the system by a device description before this device file is created.

VOL

Specifies one or more volume identifiers used by the file. The volumes must be installed in the same order as the identifiers are specified here (and as they are specified on the DEV parameter). If the file is opened for read backward, then the volume identifiers in the list are processed from last to first (while the devices in the device list are used in first-to-last order). If a list of volume identifiers is provided for the file, operator messages indicate the name of the required volume. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

This parameter overrides the volume identifiers specified in the tape device file.

***NONE:** No tape volume identifiers are specified for this file. They can be supplied before the device file is opened, either in a CHGTAPF or OVRTAPF command or in the high-level language program. If volume identifiers are not specified before the device file is opened, volume checking is not performed beyond verifying that the correct label type volume is on the device, and volume names are not provided in operator messages. The maximum number of reels processed for an *NL, *NS, *BLP, or *LTM input file when VOL(*NONE) is specified is determined by the REELS(number-of-reels) parameter value.

volume-identifier: Specify the identifiers of one or more volumes in the order in which they are placed on the device. Each volume identifier contains a maximum of 6 alphanumeric characters. Use a blank as a separator character when listing multiple identifiers. Up to 50 volume identifiers can be specified. These identifiers are used in messages sent to the operator during processing. The maximum number of reels processed for an *NL, *NS, *BLP, or *LTM input file is determined by the number of volume identifiers in the list.

Note: If the VOL parameter value used for the file specifies a list of identifiers rather than VOL(*NONE), the number-of-reels part of the REELS parameter is ignored regardless of where it is specified. A description of how the parameter values for the file are determined when overrides are used, the high-level language interface, and the device file when the file is opened is in the *Data Management Guide*. To ensure that the number-of-reels part of the REELS parameter is used (rather than a VOL identifier list) to control the volumes processed by the tape device file, specify VOL(*NONE) in the same command in which the REELS parameter is specified.

REELS

Specifies the type of labeling used on the tape reels and the maximum number of reels processed if both a list of volume identifiers is not specified (VOL parameter) and this device file is used with either *NL, *NS, *LTM, or *BLP input files. When the number of reels is specified as the second element of this parameter, the volume identifiers on the volumes are ignored if labeled tapes are being processed; instead, the order in which the reels are installed on the device must be checked by the operator.

The number-of-reels value is not a limiting value for standard-label or output files. For a standard-label *input* file, the data file labels limit the number of volumes processed by indicating end-of-file. For an *output* file, the number-of-reels value is ignored; the system requests that additional volumes be kept on the device until the file is closed.

The system checks the first record following the load point on the tape to see (1) whether it has exactly 80 bytes for EBCDIC or at least 80 bytes for ASCII and (2) whether the first 4 bytes contain the values VOL and 1. If so, the reel contains a standard-label tape. *SL and *BLP files require standard-label tape volumes. *NL, *NS, and *LTM tape files cannot process standard-label volumes.

Note: The values *SL, *NL, and *LTM can be specified if the device file is used for either reading or writing on tapes. The values *NS and *BLP are valid only if the device file is used to read tapes.

This parameter overrides the values specified in the device file, in the program, and/or in other called OVRTAPF commands.

***SL:** The volumes have standard labels. If a list of volume identifiers is specified (with the VOL parameter), the system checks that the correct tape volumes are on the device in the specified sequence.

- If no volume identifier list is given and the file is opened for *output*, any standard-label volumes may be installed on the device.
- If no volume identifier list is given and the file is opened for *input*, the first volume may have any volume identifier, but if the file is continued, the system requires the correct continuation volumes to be processed (verified by checking the data file labels). For an input file, the end-of-file message is sent to the program being used when the labels on the last volume processed indicate that it is the last volume for the data file.

***NL:** The volumes are not labeled. On a nonlabeled volume, tape marks are used to indicate the end of each data file and the end of the volume. For an *input* file, the end-of-file message is sent to the program when the number of volumes specified in the volume list have been processed, or, if no list of volume identifiers is provided, when the number of reels specified in the REELS parameter are processed.

***NS:** The volumes have nonstandard labels. Each volume must start with some kind of label information, optionally preceded by a tape marker and always followed by a tape marker. This nonstandard label information is ignored. The system spaces forward to a point beyond the tape marker that follows the nonstandard

labels and positions the tape at the file's data. Each reel must have a tape marker at the end of the file's data. Information beyond this ending tape marker is ignored. Only a single data file can exist on a nonstandard tape. Standard-label volumes *cannot* be processed by using the *NS label processing.

For an *input* file, the end-of-file message is sent to the program using the file when the number of volumes specified in the volume list have been processed, or, if no list of volume identifiers is provided, when the number of reels specified in the REELS parameter are processed.

***BLP:** Standard-label processing is bypassed. Each reel *must* have standard labels. Although each reel is checked for a standard volume label and each file must have at least one standard header label (HDR1) and one standard trailer label (EOV1 or EOF1), most other label information (such as the data file record length or block length) is ignored. The sequence number of each file on the volume is determined only by the number of tape markers between it and the start of tape (in contrast to *SL processing in which the file sequence number stored in the header and trailer labels of each file are used to locate a data file).

Most of the information in the data file trailer label is ignored, but if an end-of-file (EOF) trailer label is found, the end-of-file message is sent to the program using the tape file. If no end-of-file trailer label is encountered by the time the specified number of volumes or reels have been processed (volume identifier list and REELS parameter), the end-of-file message is immediately sent to the program using the tape file. Bypass label processing can be used when the user does not know the name of the file used or when some file label information is incorrect.

***LTM:** The volumes have no labels but do have a single leading tape marker before the first data file. REELS(*LTM) is processed the same as REELS(*NL) except that when SEQNBR(1) is specified for an output file to create the first data file on the tape, a leading tape marker is written at the start of the tape before the first data block.

number-of-reels: Specify the maximum number of reels to be processed for an *NL, *LTM, *NS, or *BLP input tape operation when a list of volume identifiers is not specified (VOL parameter). If the next reel is not on the device when the end of the currently-processing tape is reached, a message is sent to the operator requesting that the next tape be installed on the next tape device. The number-of-reels value is ignored for a standard-label (*SL) file or for any output file.

SEQNBR

Specifies the sequence number of the data file on the tape being processed.

- When standard-label tapes are used, the four-position file sequence number is read from the first header label of the data file.
- When bypass label processing is used or when standard-label tapes are not used, the system counts the tape markers from the start of the tape to locate the correct sequence number data file to be processed.
- When multiple-file, multiple-volume tapes are processed using REELS(*SL), the file sequence numbers continue consecutively through the volumes; thus, each new data file has a sequence number one greater than the previous file, regardless of its volume location.

***END:** The file is written on the end of the tape. This value is used only for files that are written to tape.

An error message is shown on the display when a tape device file is used to read from a tape and the *END special value is specified in the tape device file.

***NEXT:** The next file in the sequence is processed. This value is used for files read from tape. If the tape is currently in a position that is prior to the first file, the first file on the tape is processed.

An error message is shown on the display when a tape file is used to write to a tape and the *NEXT special value is specified in the tape file.

sequence-number: Specify the sequence number of the file.

LABEL

Specifies the data file identifier of the data file processed by this tape device file. An identifier is defined only for standard-label tapes and is stored in the header label immediately before the data file.

If a data file identifier is specified for any type of label processing other than *SL, it is ignored.

An identifier is required for a standard label output file, but is optional for an input file because the sequence number uniquely identifies the data file to process.

For an input file or output file with EXTEND(*YES) specified, this parameter specifies the identifier of the data file on the tape. The specified identifier must match the one in the labels of the data file that the SEQNBR parameter specifies; otherwise, an error message is sent to the program using this device file. For output files with EXTEND(*NO) specified, this parameter specifies the identifier of the data file to be created on the tape. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

This parameter overrides the data file identifier specified in the device file, in the program, and/or in other called OVRTAPF commands.

data-file-identifier: Specify the identifier (17 alphanumeric characters maximum) of the data file used with this tape device file. If this identifier is for a tape written

in the basic exchange format, and is used on a system other than an AS/400 system, up to eight characters or a qualified identifier having no more than eight characters per qualifier must be used.

RCDLEN

Specifies, in bytes, the length of the records contained in the data file processed with this device file. The system always uses the record length and block length specified in the data file labels for any standard-label input file or output file with EXTEND(*YES) specified (if a second header label (HDR2) is found on the tape and *BLP label processing has not been specified).

This parameter overrides the value specified in the device file, in the program, and/or in other called OVRTAPF commands.

***CALC:** No record length is specified for the data file being processed. If *CALC is specified, the system will attempt to calculate an appropriate record length when the file is opened. RCDLEN(*CALC) can be used for nonlabeled tapes or when there is no HDR2 label if a BLKLEN value other than *CALC is specified for the file and RCDBLKFMT does not specify spanned or blocked records. In this case, the system calculates an appropriate record length from the block length, record block format, and buffer offset (for an ASCII file) specified for the file. In any other case, the actual record length must be specified by a CHGTAPF command or OVRTAPF command, or in the high-level language program that opens the device file.

record-length: Specify a value ranging from 1 through 32767 bytes that indicates the length of each record in the data file. The minimum and maximum record length allowed for a file is dependent on the record block format, block length, buffer offset (for an ASCII file), and recording code. The *Record Length Values* table (at the end of this command description) shows the minimum and maximum record length values allowed for each record block format, assuming the block length value is large enough to support the maximum record length.

BLKLEN

Specifies, in bytes, the maximum length of the data blocks transferred to or from the tape for output or input operations. The system uses the block length and record length specified in the data file labels for any standard-label input file or output file with EXTEND(*YES) specified (if a second header label (HDR2) is found on the tape and *BLP label processing has not been specified).

This parameter overrides the value specified in the device file, in the program, or in other OVRTAPF commands.

***CALC:** No block length is specified for the data file to be processed. If *CALC is specified, the system attempts to calculate an appropriate block length when the file is opened. BLKLEN(*CALC) can be used for nonlabeled tapes or when there is no HDR2 label if a RCDLEN value other than *CALC is specified for the file and RCDBLKFMT does not specify spanned or blocked records. In this case, the system calculates an appropriate block length from the record length, record block format, and buffer offset (for an ASCII file) specified for the file. In any other case, the actual block length must be specified by a CHGTAPF command or OVRTAPF command, or in the high-level language program that opens the device file.

block-length: Specify a value, not exceeding 32767 bytes, that specifies the maximum length of each block in the data file to be processed. The minimum block length that can be successfully processed is determined by the tape device hardware and AS/400 system machine support functions.

- The maximum block length is always 32767 bytes for an input file, but is limited to 9999 bytes if block descriptors must be created for an ASCII output file.
- The following table shows the minimum and maximum block length values allowed for an output file:

Table 62. Absolute Block Length (BLKLEN) Ranges

CODE	BUFOFSET	Minimum BLKLEN	Maximum BLKLEN
*EBCDIC	Ignored	18	32767
*ASCII	0	18	32767
*ASCII	*BLKDSC	18	9999

BUFOFSET

Specifies the buffer offset value for the start of the first record in each block in the tape data file. A buffer offset value can be used for any record block format ASCII file, and is ignored for an EBCDIC tape file. The system uses the buffer offset specified in the data file labels for any standard-label input file or output file with EXTEND(*YES) specified if a value is contained in the second header label (HDR2) on the tape, and *BLP label processing has not been specified.

The buffer offset parameter specifies the length of any information that precedes the first record in the block. For record block formats *D, *DB, *VS, and *VBS, each record or record segment is preceded by a descriptor that contains the length of the record or segment. A buffer offset value is used to indicate that there is information *ahead* of the descriptor word for the first record in each block, or *ahead* of the data of the first fixed-length record or undefined format record in each block.

This parameter is not needed for a standard-label file processed for input if the tape includes a second file header label (HDR2) that contains the buffer offset value. A buffer offset value must be provided by the Create Tape File (CRTTAPF) command, Change Tape File (CHGTAPF) command, or Override Tape File (OVRTAPF) command, or by the file labels for an input file that contains any information (such as a block

descriptor) ahead of the first record in each block. If the user does not specify a buffer offset value when a tape file is created, it is not necessary to specify an offset value when the file is read.

The only buffer offset values allowed for an output file are zero and *BLKDSC. An existing standard-label data file with a buffer offset value in the HDR2 label can be extended only if the buffer offset value is either 0 or 4. A buffer offset value of 0 in the HDR2 label adds data blocks with *no* buffer offset. BUFOFSET(*BLKDSC) must be specified to extend an existing tape data file that contains an offset value of 4 in the HDR2 label.

This parameter overrides the value specified in the device file, in the program, and/or in other called OVRTAPF commands.

***BLKDSC:** Creates 4-byte block descriptors in any tape file created by using this device file. Any input file read by using this device file should assume 4 bytes of buffer offset information preceding the first record in each data block. This value is valid only for a record block format *D or *DB file. The contents of the buffer offset information of each output data block when BUFOFSET(*BLKDSC) is specified is the actual length of the data block, expressed in zoned decimal format.

buffer-offset: Specify a value ranging from 0 through 99 bytes that specifies the length of the buffer offset information that precedes the first record in each data block.

RCDBLKFMT

Specifies the type and blocking attribute of records in the tape data file being processed.

Record block format *V and *VB records can be processed only for an EBCDIC file; *D and *DB records can be processed only for an ASCII file. If a standard-label tape (label type *SL or *BLP) is being processed and an inconsistent record block format is specified for the volume code, the correct record type is assumed (V or D) for the volume code and a warning message is sent to the program that opens the file. If the record type and code are inconsistent for a nonlabeled volume (label type *NL, *LTM, or *NS), an error message is sent and the file is *not* opened, because there are no labels to verify the correct volume code.

If a valid record length, block length, and buffer offset value (for an ASCII file) are specified for fixed-length records but the block attribute is incorrect, the correct block attribute is assumed (changing record block format *F to *FB or record block format *FB to *F), and a warning message is sent to the program that opens the file.

If a block length is specified that is longer than required to process a maximum length record, then record block format *V, *D, or *VS is changed to *VB, *DB, or *VBS and a warning message is sent to the program that opens the file.

The *Required RCDLEN/BLKLEN/BUFOFSET Relation* table, at the end of this command description, shows the required relationship between the record length, block length, and buffer offset (for ASCII) file parameters for an output file or an input file where the file parameters are not determined from a second file header label (HDR2).

Note: When BUFOFSET(*BLKDSC) is specified for the file, a value of 4 should be used for the BUFOFSET part of any BLKLEN calculations, unless existing file labels on the tape specify a different value.

This parameter overrides the value specified in the device file, in the program, and/or in other called OVRTAPF commands.

***F:** Fixed-length, unblocked, unspanned records in either EBCDIC or ASCII code are processed. The system may change this record block format to *FB, based on other file parameters.

***FB:** Fixed-length, blocked, unspanned records in either EBCDIC or ASCII code are processed. The system may change this record block format to *F, based on other file parameters.

***V:** Variable-length, unblocked, unspanned records in EBCDIC type V format are processed. The system may change this record block format to *VB, *D, or *DB, based on other file parameters.

***VB:** Variable-length, blocked, unspanned records in EBCDIC type V format are processed. The system may change this record block format to *DB, based on the volume code.

***D:** Variable-length, unblocked, unspanned records in ASCII type D format are processed. The system may change this record block format to *DB, *V, or *VB, based on other file parameters.

***DB:** Variable-length, blocked, unspanned records in ASCII type D format are processed. The system may change this record block format to *VB, based on the volume code.

***VS:** Variable-length, unblocked, spanned records in either EBCDIC or ASCII code are processed. The system may change this record block format to *VBS, based on other file parameters. Note that the representation of spanned records on the tape is different for EBCDIC and ASCII files, but the system selects the correct format based on the file code.

***VBS:** Variable-length, blocked, spanned records in either EBCDIC or ASCII code are processed. Note that the representation of spanned records on the tape differs for EBCDIC and ASCII files, but the system selects the correct format based on the file code.

***U:** Undefined format records in either EBCDIC or ASCII code are processed. RCDBLKFMT(*U) records are processed as variable-length records, and each record written or read is in a separate tape block. This format can be useful for processing tape files that do not have the formatting requirements of any other record

block format. or ASCII code. RCDBLKFMT(*U) records are processed as variable length records, where each record written or read is in a separate tape block. This format is useful for processing tape device files that do not meet the formatting requirements of any other record block
format.

EXTEND

Specifies, for output operations to tape, whether new records are added to the end of a data file currently on the tape. The specific data file is identified by the SEQNBR parameter and, for a standard-label file, the LABEL parameter. If the data file is extended, it becomes the last file on the tape volume; data files that follow it are overwritten as the specified file is extended.

Note: This parameter is not valid for 1/4-inch cartridge tape devices.

This parameter overrides the extend value specified in the device file, in the program, and/or in other called OVRTAPF commands.

Element 1: Adding Records to Data File

***NO:** Records are not added to the end of the specified data file. If there is already a data file with the specified SEQNBR on the tape, a new data file is created by overwriting the existing data file and any files that follow it. Records are not added to the end of the specified data

***YES:** New records are added to the end of the specified data file on tape when this device file is used.

Element 2: Checking Active Files

If EXTEND(*YES) is specified, the following values check to see whether the file already exists:

***NOCHECK:** The file is extended without being checked to see whether the file is active.

***CHECK:** Before the file is extended, the file is checked to see whether it is active.

DENSITY

Specifies, in bits per inch, the density of the data that is written on the tape volume when this device file is created. This parameter is used only for tape files being written to tape; it is ignored for tape files being read from the tape (in the case of files being read from tape, the density on the tape is used).

The density of a standard-label volume is specified on the INZTAP command, which initializes tapes as standard-label volumes by writing volume labels on them. If the density specified on this parameter is different than the density of a standard-labeled tape, the density specified on this parameter is used, and a warning message is sent.

***DEVTYPE:** The density is based on the kind of tape device being used.

1600: The data density on this tape volume is 1,600 bits per inch.

3200: The data density on this tape volume is 3,200 bits per inch.

6250: The data density on this tape volume is 6,250 bits per inch.

43200: The data density on this tape volume is 43,200 bits per inch.

***FMT3480:** The data density on this tape volume is formatted to support a 3480 device.

***FMT3490E:** The data density on this tape volume is formatted to support a 3490 device.

***QIC120:** The format of this tape is *QIC120.

***QIC525:** The format of this tape is *QIC525.

***QIC1000:** The format of this tape is *QIC1000.

***FMT2GB:** The format of this tape is *FMT2GB.

***FMT5GB:** The format of this tape is *FMT5GB.

CODE

Specifies the character code used. The code can be either extended binary-coded decimal interchange code (*EBCDIC) or the American National Standard Code for Information Interchange (*ASCII).

***EBCDIC:** The extended binary-coded decimal interchange code (EBCDIC) character set code is used.

***ASCII:** The ASCII character set code is used.

CRTDATE

Specifies, for tape input data files and for tape output for which EXTEND(*YES) is specified, the date when the data file was created (written on tape).

Note: The data file creation date is stored in file labels on the tape. If a creation date is specified for any type of label processing other than standard-label (*SL), it is ignored. If the creation date written on the tape containing the data file does not match the date specified in this device file description, an inquiry message is sent to the operator.

This parameter overrides the value specified in the program, device file, and/or in other called OVRTAPF commands.

***NONE:** The creation date is not specified. It is not checked unless it is supplied before the device file is opened, either in a OVRTAPF command or CHGTAPF command, or in the high-level language program.

creation-date: Specify the creation date of the data file used by this device file. The date must be specified in the format defined by the job attributes DATFMT and, if separators are used, DATSEP.

EXPDATE

Specifies, for tape output data files only, and only when standard-labeled tapes are used, the expiration date of the data file used by this device file. If a date is specified, the data file is protected and cannot be overwritten

until after the specified expiration date. The files cannot be overwritten until after the expiration date.

Note: The data file expiration date is stored in file labels on the tape. If an expiration date is specified for any type of label processing other than *SL, it is ignored.

This parameter overrides the value specified in the program, device file, and/or in other called OVRTAPF commands.

***NONE:** No expiration date for the data file is specified; the file is not protected. An expiration date is written in the data file labels so the file can be used as a scratch data file.

***PERM:** The data file is permanently protected. An expiration date of 999999 is assigned.

expiration-date: Specify the date on which the data file expires, after which it can be overwritten with new data. The expiration date must be later than or equal to the current date. The date must be specified in the format defined by the job attributes QDATFMT and, if separators are used, QDATSEP.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

Note: Unless an ending option is specified by the high-level language program when the file is closed, this parameter overrides the ending operation specified in the device file, in the program, and/or in other called OVRTAPF commands.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

This option is used to reduce the time required to position the tape if the next tape file that opens to this device uses a data file that is on this volume.

Note: Even if ENDOPT(*LEAVE) is specified, the next tape file opened to this reel is positioned at the beginning of some data file on the volume (or at the end of a data file, for either read backward or for output that extends an existing data file on the volume). A tape file is always positioned at the start or end of a data file when it is opened.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

USRLBLPGM

Specifies the qualified name of the user program that processes user tape labels. On an output file, the user label program will pass the user labels that are written to tape. On an input file, the user labels are passed to the user label program.

***NONE:** There is no user label program for this device file.

The name of the user label program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

user-label-program-name: Specify the name of the user program that processes the user tape labels. If no library qualifier is given, *LIBL is used to find the file.

IGCDTA

Specifies whether the file processes double-byte character set (DBCS) data.

***NO:** The file does not process DBCS data.

***YES:** The file processes DBCS data.

WAITFILE

Specifies the number of seconds that the program waits for the file resources and session resources to be allocated when the file is opened, or for the device or session resources to be allocated when an acquire operation is performed to the file. If those resources are not allocated within the specified wait time, an error message is sent to the program. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: An immediate allocation of the device by the device resource is required when an acquire operation is performed to the file.

This parameter overrides the wait time specified in the program or in the device file.

***IMMED:** The program does not wait; when the file is opened, an immediate allocation of the file resources is required.

***CLS:** The job default wait time is used as the wait time for the file resources being allocated.

number-of-seconds: Specify the number of seconds that the program waits for the file resources to be allocated to the tape file when the file is opened, or the wait time for the device allocated when an acquire operation is performed to the file. Valid values range from 1 through 32767 seconds.

SECURE

Specifies whether this file is safe from the effects of previously called file override commands. If SECURE is not specified, processing occurs as if SECURE(*NO) is specified.

***NO:** This file is not protected from the effects of other file overrides; its values can be overridden by the effects of previously called file override commands.

***YES:** This file is protected from the effects of any file override commands previously called.

OVRSCOPE

Specifies the extent of influence (scope) of the override.

***CALLLVL:** The scope of the override is determined by the current call level. All open operations done at a call level that is the same as or higher than the current call level are influenced by this override.

***JOB:** The scope of the override is the job in which the override occurs.

SHARE

Specifies whether the open data path (ODP) for the tape file is shared with other programs in the routing step. When an ODP is shared, the programs accessing the file share facilities such as the file status and the buffer.

More information on shared database files is in the *Database Guide*.

***NO:** The ODP created by the program with this attribute is not shared with other programs in the routing step. Every time a program opens the file with this attribute, a new ODP to the file is created and activated.

Note: This includes many open files in the same program.

***YES:** The ODP created with this attribute is shared with each program in the routing step that also specifies SHARE(*YES) when it opens the file.

Note: When SHARE(*YES) is specified and control is passed to a program, a read operation in that program retrieves the next input record. A write operation produces the next output record.

OPNSCOPE

Specifies the extent of influence (scope) of the open operation.

***ACTGRPDFN:** The scope of the open operation is determined by the activation group of the program that called the OVRTAPF command processing program. If the activation group is the default activation group, the scope is the call level of the caller. If the activation group is a non-default activation group, the scope is the activation group of the caller.

***JOB:** The scope of the open operation is the job in which the open operation occurs.

Table 63. Record Length Values

Absolute RCDLEN Ranges					
		FILETYPE(*DATA)		FILETYPE(*SRC)	
CODE	RCDFBLKFMT	Minimum RCDLEN	Maximum RCDLEN	Minimum RCDLEN	Maximum RCDLEN
*EBCDIC *ASCII	*F *FB *U *F *FB *U	18 18	32767 32767	30 30	32767 32767
*EBCDIC *ASCII	*V *VB *D *DB	1 1	32759 9995	13 13	32767 10007
*EBCDIC *ASCII	*VS *VBS *VS *VBS	1 1	32759 32759	13 13	32767 32767

Table 64. Required RCDLEN/BLKLEN/BUFOFSET Relation

CODE	RCDBLKFMT	BLKLEN = fcn(RCDLEN,BUFOFSET)
*EBCDIC *ASCII	*F *U *F *U	BLKLEN = RCDLEN BLKLEN=RCDLEN + BUFOFSET
*EBCDIC *ASCII	*FB *FB	BLKLEN = RCDLEN * n BLKLEN = (RCDLEN * n) + BUFOFSET n is the number of records in a maximum-length block
*EBCDIC *ASCII	*V *D	BLKLEN = RCDLEN * 8 BLKLEN = RCDLEN * 4 + BUFOFSET
*EBCDIC *ASCII	*VB *DB	BLKLEN >= RCDLEN + 8 BLKLEN >= RCDLEN + 4 + BUFOFSET
*EBCDIC *ASCII	*VS *VBS *BS *VBS	BLKLEN >= 18 BLKLEN >= 6 + BUFOFSET (18 minimum)

Examples

Example 1: Overriding a File

```
OVRTAPF  FILE(OUT)  VOL(DPT706)  LABEL(STATUSR)
```

This command overrides a file named OUT in the program using the data file STATUSR on tape volume DPT706.

Example 2: Allowing DBCS Data

```
OVRTAPF  FILE(IGCLIB/IGCTAP)  IGCDTA(*YES)
```

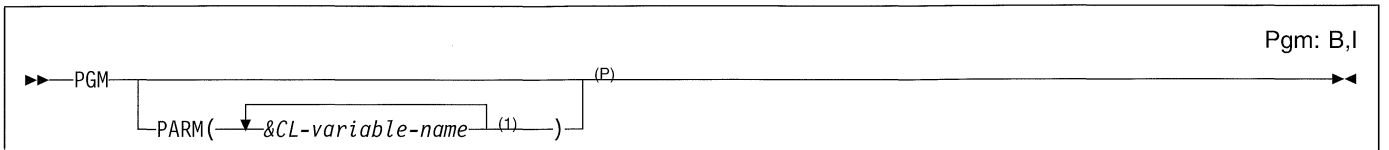
This command overrides the tape device file named IGCTAP, which is stored in the library IGCLIB, so the file may contain double-byte character set data.

Example 3: Using Data Density of 1600 Bits Per Inch

```
OVRTAPF  FILE(OUT)  DENSITY(1600)
```

This command overrides a file named OUT to use a data density of 1600 bits per inch when writing to the tape volume.

PGM (Program) Command



Notes:
1 A maximum of 40 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Program (PGM) command is used in a CL program source file to identify the start of a CL program that is to be compiled and to specify the parameters that are to be received by the program after it is compiled. If a PGM command is used, it must be the first command in the program source file; if a PGM command is not used, a PGM command without parameters is assumed. The name of the program is specified in the Create Control Language Program (CRTCLPGM) command that is used to create the CL program.

The PGM command also specifies any parameters that are passed to the program when it is called for processing by another program. For information about how constants are passed, see the PARM parameter description under the CALL command.

This program can be called by a Call (CALL) or Transfer Control (TFRCTL) command, or by a routing entry in a subsystem description. When the program is called by a CALL or TFRCTL command, the specified parameters can be passed to it.

Parameters defined in this command must be passed when the program is called. The parameters passed must be of the type, length, and order specified in this command. Each of the parameter values passed to this program can be a character string, a numeric value, or a CL variable. When received by this program, each value is given a different CL variable name. Each CL variable name must be defined in the CL source file by a separate DCL (Declare) command before the program is compiled. Up to 40 parameters can be passed.

Restriction: This command is valid only in a CL program.

Optional Parameters

PARM

Specifies the names of one or more CL variables that are to receive the parameter values passed to this program. Specify a CL variable name for each of the values to be received from the calling program; the name must start with an ampersand (&). Null values, *N, cannot be specified for any parameter. The parameter values are associated with the variables in the PARM parameter in the order in which they were specified on the CALL or TFRCTL commands. The type and length of each value passed must have matching attributes in the calling and receiving programs. However, for character constants, the receiving program can specify a shorter length; when this is done, the character string passed is truncated to the length declared in the receiving program. For information on how each data type is passed, see the description of the PARM parameter in the CALL command.

Examples

Example 1: Program Containing No Parameters

```
PGM
  •
  •
  •
ENDPGM
```

This PGM command is the first command in a CL program source file for a program that contains no parameters.

Example 2: Program Containing Two Parameters

```
PGM   PARM(&X &Y)
```

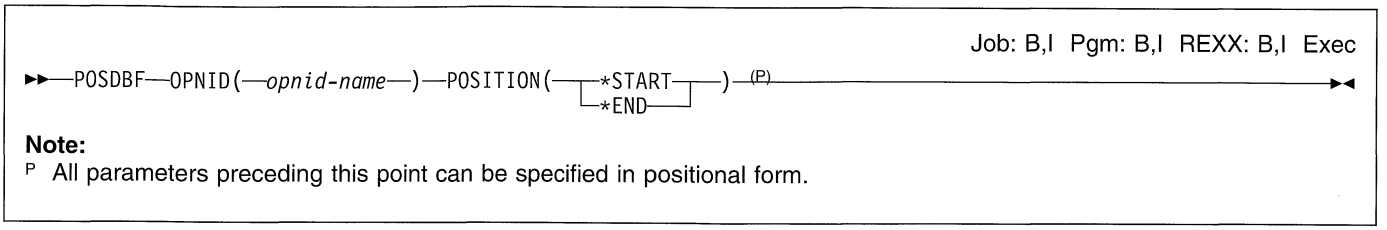
This is the first command in a program source file for a program that contains two parameters, &X and &Y, that have values passed to them from the calling program.

Example 3: Program Containing Two Parameters in Positional Form

```
PGM   (&PARM1 &PARM2)
```

This is the first command in a program source file for a program that specifies two parameters in positional form, &PARM1 and &PARM2. When this program is called, the calling program passes, through the CALL or TFRCTL command, the values to be assumed for &PARM1 and &PARM2.

POSDBF (Position Database File) Command



Purpose

The Position Database File (POSDBF) command allows the user to set the position of a database file to either the beginning or end of an open file.

Required Parameters

OPNID

Specifies which opened file changes position. This file must be opened by either the Open Database File (OPNDBF) or Open Query File (OPNQRYF) command.

POSITION

Specifies the starting or ending position of the database file.

***START:** The position of the database file is set to the start position of the member currently open. After the start position is set, a read next operation gets the first record in the current member. If the Override Database File (OVRDBF) command MBR(*ALL) processing is in effect, a read previous operation gets the last record in the previous member. If a read previous is done and the previous member does not exist, the end of the file message (CPF5001) is sent.

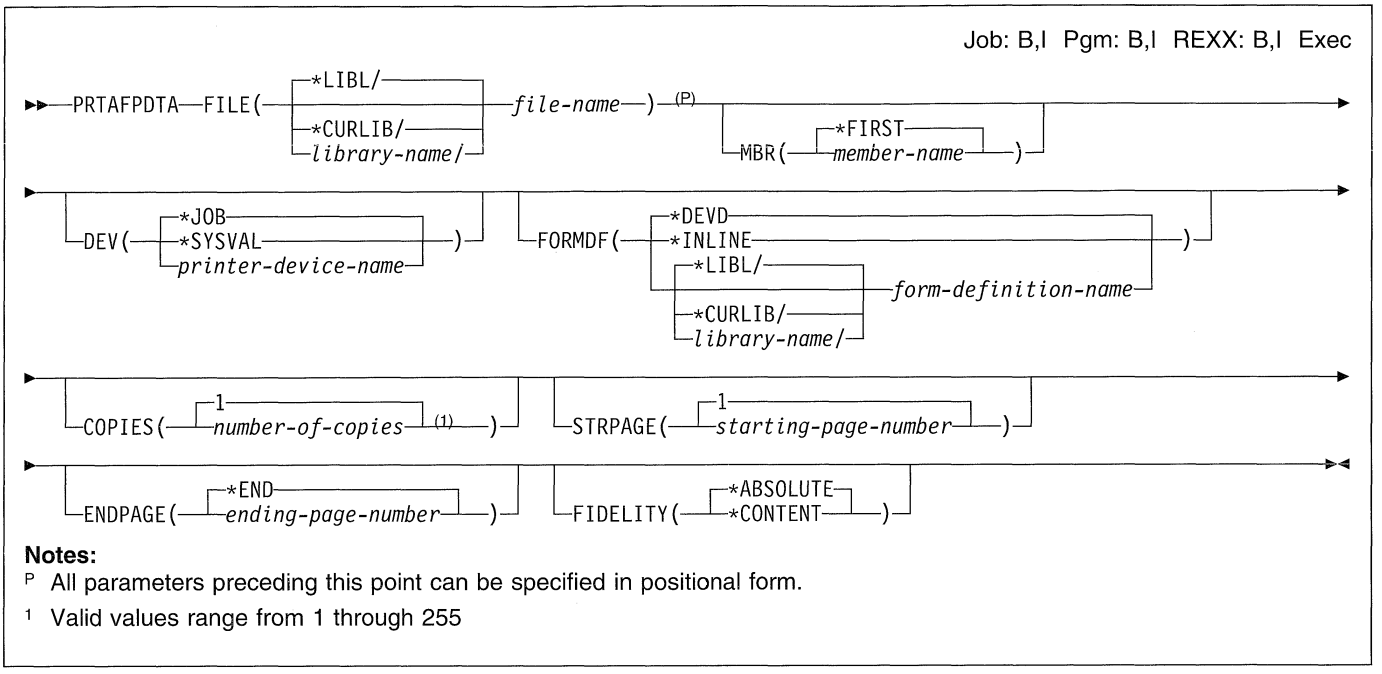
***END:** The position of the database file is set to the end of the member that is currently open. After the end position is set, a read previous operation gets the last record in the current member. If the Override Database File (OVRDBF) command MBR(*ALL) processing is in effect, a read next operation gets the first record in the next member. If a read next is done and the next member does not exist, the end of the file message (CPF5001) is sent.

Example

```
POSDBF  OPNID(XXX)  POSITION(*START)
```

This command sets the record position of the database file that is opened with OPNID(XXX) to the starting position of the database file member that is currently open.

PRTAFPDTA (Print Advanced Function Printer Data) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 Valid values range from 1 through 255

Purpose

The Print Advanced Function Printer Data (PRTAFPDTA) command prints output received from a System/370* host. This command provides the user with a means of specifying the file being printed and the parameters used to control the print operation.

Required Parameters

FILE

Specifies the qualified name of the Advanced Function Printing Data Stream (AFPDS) file to be printed. Only physical files are supported for this command. If you use the Override with Printer File (OVRPRTF) command with PRTAFPDTA, do not override the device type (DEVTYPE parameter).

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the AFPDS to be printed.

Optional Parameters

MBR

Specifies the member that contains the data to be printed.

***FIRST:** The first member in the database file is used.

member-name: Specify the member that contains the data to be printed.

DEV

Specifies the printer that prints the file.

***JOB:** The printer device specified in the job description is used.

***SYSVAL:** The value specified in the system value QPRTDEV is used.

printer-device-name: Specify the name of the printer.

FORMDF

Specifies the form definition to use when printing the file. A form definition is a resource object that defines the characteristics of the form, including: overlays, position of page data on the form, and number of copies of pages and modification to pages. The form definition is located inline with the file being printed, or in a library.

***DEVD:** The device description obtains the name of the form definition being used. If no value is specified, *DEVD is assumed.

***INLINE:** The form definition that is inline with the file that is printed is used.

The name of the form definition can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

form-definition-name: Specify the name of the form definition that must exist in the library named. A valid name has up to 8 characters.

COPIES

Specifies, for spooled files, the number of copies being printed.

1: One copy of the output is printed.

number-of-copies: Specify the number of copies that are printed.

STRPAGE

Specifies the page on which printing starts. This parameter is used for partial printing of a file.

1: Printing starts with page 1. If the start page is not specified, 1 is assumed.

starting-page-number: Specify the page number on which printing starts.

ENDPAGE

Specifies the page on which printing ends. This parameter is used for partial printing of a file ending at a specified page number. If both start page and end page are specified, the end page must be greater than or equal to the start page. Specifying an end page beyond the end of the actual file does not create an error condition.

***END:** Printing concludes at the end of the file.

ending-page-number: Specify the page number on which printing ends.

FIDELITY

Specifies the degree of exactness required when printing the file.

***ABSOLUTE:** The job is printed only if the file can be printed exactly as specified by the data stream and external controls.

***CONTENT:** Prints the file using all available exception handling.

Examples

Example 1: Printing Specific Pages

```
PRTAFPDTA  FILE(MYLIB/MYFILE)
  STRPAGE(2)  ENDPAGE(6)
```

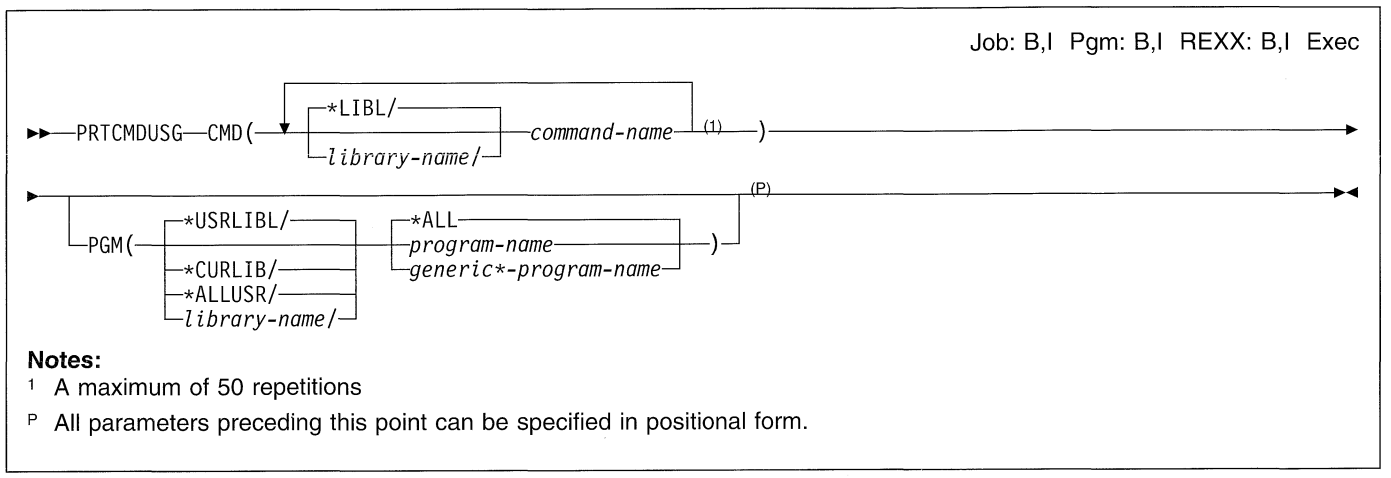
This command prints the first member in file MYFILE in library MYLIB starting with page 2 and ending on page 6.

Example 2: Printing Using All Available Exception Handling

```
PRTAFPDTA  FILE(MYLIB/MYFILE)
  FORMDF(F10101)  FIDELITY(*CONTENT)
```

This command prints the first member in file MYFILE in library MYLIB using a form definition of F10101 and all available exception handling.

PRTCMDUSG (Print Command Usage) Command



Purpose

The Print Command Usage (PRTCMDUSG) command creates a cross-referenced list of a specified group of CL commands that are used in a specified group of CL programs. The report shows, program by program, which of the specified commands are used in each program. The report can be used to identify which programs need to be recompiled because of changes that have been made to the command definition objects of commands specified on the PRTCMDUSG command. Note that this command can take a long time to run and can make a lot of printed output.

Required Parameters

CMD

Specifies the qualified names of up to 50 CL commands for which specified programs are searched and printed in a command usage report. The system searches the specified programs for every occurrence of each library-name/command-name character string you specify.

The name of the command can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

library-name: Specify the name of the library to be searched.

command-name: Specify the names of the commands for which specified programs are searched.

Optional Parameters

PGM

Specifies the qualified name of the program or the generic name of several programs that are searched for the specified commands. Only the programs and libraries for which the user has some (any) authority are included in the report. This parameter also can specify that all (*ALL) programs in the specified library or libraries (*USRLIBL/*ALLUSR, for example) are searched.

The name of the program can be qualified by one of the following library values:

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All CL programs in the specified library (or all libraries identified in the library qualifier) for which the user has some authority are searched.

program-name: Specify the name of the program being searched for the specified command.

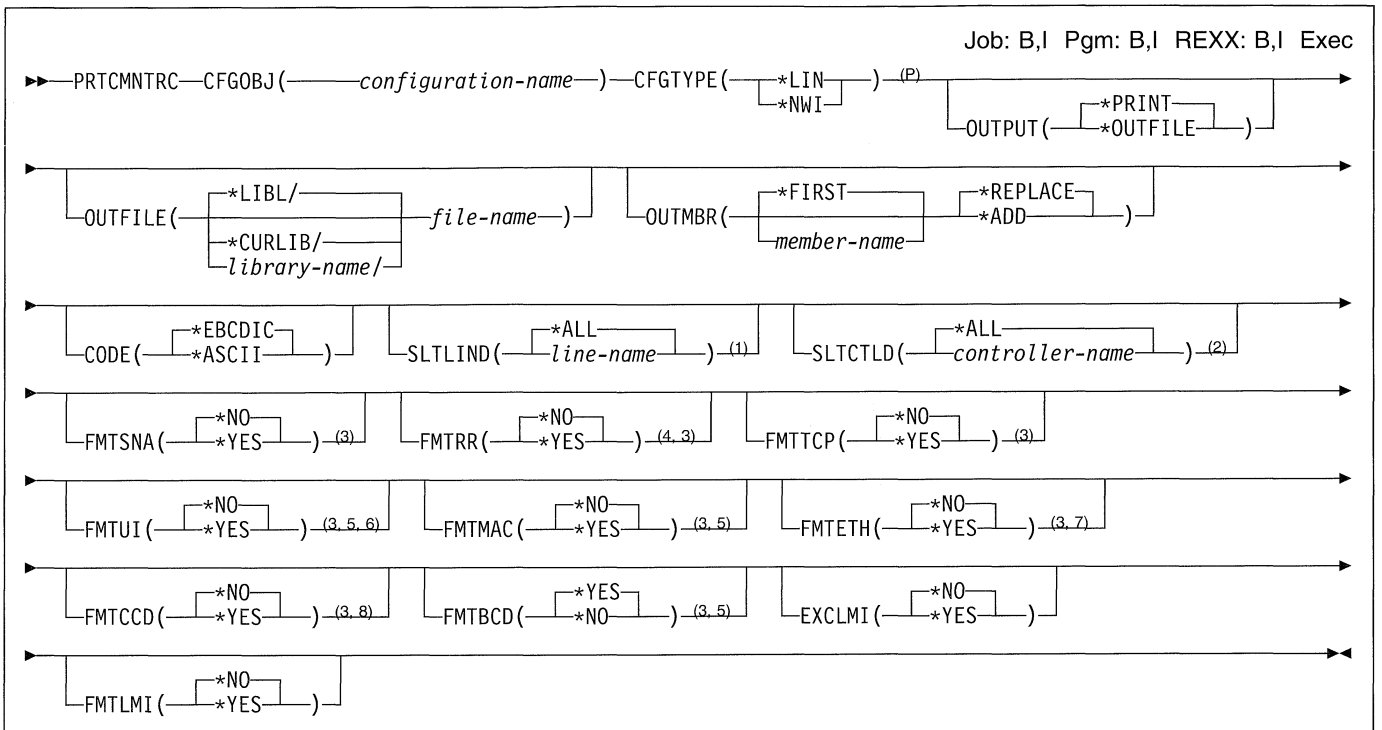
generic-program-name:* Specify the name of the program or the generic name of several programs in the specified library qualifier that are searched for the specified commands. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be printed only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Example

```
PRTCMDUSG   CMD(CPYF)   PGM(PAYROLL/*ALL)
```

This commands searches all CL programs in the library PAYROLL for the Copy File (CPYF) commands and prints the names of both the commands and the program.

PRTCMNTRC (Print Communications Trace) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 This parameter is valid only if CFGTYPE(*NWI) is specified.

2 This option is valid only for the following controllers: ASYNC, X.25, ISDN data link control (IDLC), synchronous data link control (SDLC), and local area network (LAN)

3 This parameter is not used when printing to an output file.

4 This option is valid only for LANs, SDLC, IDLC, ISDN, and X.25.

5 This option is valid only for LANs.

6 FMTSNA(*NO) must be specified if FMTUI(*YES) is specified.

7 This option is valid only on Ethernet networks.

8 This option is valid only for the object type of *NWID.

Purpose

The Print Communications Trace (PRTCMNTRC) command transfers the communications trace data for the specified line or network interface description to a spooled file or an output file.

Restrictions:

1. *SERVICE authority is required to use this command.
2. This command is shipped with public *EXCLUDE authority.
3. The following user profiles have authority to this command:
   - QSECOFR
   - QSRV

Required Parameters

CFGOBJ
Specifies the name of the configuration object being traced. The object is either a line description or a network interface description.

CFGTYPE
Specifies the type of configuration description being traced.

***LIN:** The type of configuration object is a line description.

***NWI:** The type of configuration object is a network interface description.

Optional Parameters

OUTPUT
Specifies the format of the output data.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

OUTFILE
Specifies the database file where the output file is stored.

The name of the output file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the output file name.

OUTMBR
Specifies the name of the database file member to which the output is directed.

Element 1: Member to Receive Output

***FIRST:** The first member in the database file is used.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

CODE
Specifies the character code used. The code can be either extended binary-coded decimal interchange code (*EBCDIC) or the American National Standard Code for Information Interchange (*ASCII).

***EBCDIC:** Format the user data in EBCDIC.

***ASCII:** Format the user data in ASCII.

SLTLIND
Specifies whether to format data for all lines or a specific line communicating on the network during a trace.

***ALL:** Formats the data for all lines.

line-name: Specify the line name.

SLTCTLD
Specifies whether to format data for all controllers or a specific controller communicating on the network during a trace.

***ALL:** Formats data for all controllers.

controller-name: Specify the controller name.

FMTSNA
Specifies whether line protocol data or Systems Network Architecture (SNA) data is formatted. Line protocol data includes SDLC, X.25, Carrier Sense Multiple Access with Collision Detection (CSMA/CD), and IBM Token-Ring Network (TRLAN).

***NO:** Only line protocol data is formatted.

***YES:** Only SNA data is formatted.

FMTRR
Specifies whether receiver ready (RR) and receiver not ready (RNR) commands are formatted with other data.

***NO:** RR and RNR commands are not formatted with other data.

***YES:** RR and RNR commands are formatted with other data.

FMTTCP
Specifies whether line protocol data or Transmission Control Protocol/Internet Protocol (TCP/IP) data is formatted.

***NO:** Only line protocol data is formatted.

***YES:** Only TCP/IP data is formatted.

FMTUI
Specifies whether line protocol data or unnumbered information (UI) data is formatted.

***NO:** All line protocol data is formatted.

***YES:** Only UI data is formatted.

FMTMAC
Specifies whether line protocol data or medium access control (MAC) or station management (SMT) data is formatted.

***NO:** The line protocol data is formatted.

***YES:** Only MAC or SMT data is formatted.

FMTETH
Specifies whether IEEE 802.3 data or Ethernet V2 data is formatted.

***NO:** Only IEEE 802.3 data is formatted.

***YES:** IEEE 802.3 data and Ethernet V2 data are formatted.

FMTCCD
Specifies whether all network interface data or only Integrate Services Digital Network (ISDN) signalling data is formatted.

***NO:** All network interface data is formatted.

***YES:** Only Integrate Services Digital Network (ISDN) signaling data is formatted.

FMTBCD
Specifies whether broadcast data and data received containing destination MAC addresses is formatted.

***YES:** Broadcast data is formatted.

***NO:** Broadcast data is not formatted.

EXCLMI
Specifies whether to exclude Local Management Interface (LMI) data from the formatted output.

***NO:** LMI data is not excluded from the formatted output.

***YES:** LMI data is excluded from the formatted output.

FMTLMI
Specifies whether LMI data is formatted for the output data.

***NO:** LMI data is not formatted.

***YES:** LMI data is formatted.

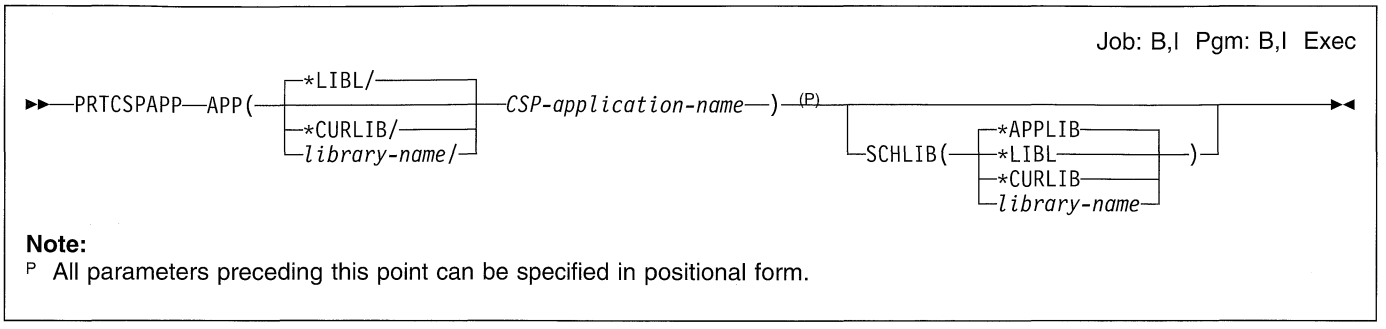
Note: You cannot specify *YES on both the EXCLMI and FMTLMI parameters.

Example

```
PRTCMNTRC  CFGOBJ(*QESLINE)  CFGTYPE(*LIN)
```

This command prints communications trace data for line description QESLINE.

PRTCSPAPP (Print CSP/AE Application) Command



Note:
(P) All parameters preceding this point can be specified in positional form.

Purpose

The Print CSP/AE Application (PRTCSPAPP) command provides a list of the names of AS/400 objects that the specified Cross System Product/Application Execution (CSP/AE) application requires to run. The command also checks for the existence of each object that the application requires. When searching for the objects that the application requires, the command searches through the same libraries that the application uses when it runs. The user can change this set of libraries by directing the command to search the library list, the current library, or an explicitly named library. The list of the names of objects required by the application is sent to the current job's output queue.

The AS/400 objects that may be referred to by a CSP/AE application are:

- Other CSP/AE applications or high-level language (HLL) programs (*PGM)
- CSP/AE map groups (*CSPMAP)
- CSP/AE data tables (*CSPTBL)
- Message files (*MSGF)
- Data files (*FILE).

The list of object names is placed in a spooled printer file, QPAEAPP, and the spooled printer file is put in the current job's output queue. If *APPLIB or *LIBL is specified as the search library, then the name of each library in the library list is included in the printer file.

Required Parameters

APP

Specifies the qualified name of a CSP/AE application, and the library where the application is stored.

The name of the object can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

CSP-application-name: Specify the name of the CSP/AE application to be processed.

Optional Parameters

SCHLIB

Specifies the libraries to search for the objects used by the specified application.

***APPLIB:** The library value the application uses at run time is used to qualify objects used by the application.

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

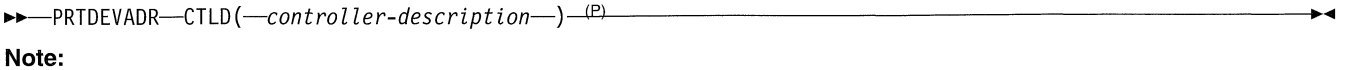
Example

```
PRTCSPAPP   APP(PAYROLL)
```

This command creates a spooled printer file showing a list of all the objects to which the application PAYROLL refers.

PRTDEVADR (Print Device Addresses) Command

Job: B,I Pgm: B,I REXX: B,I Exec



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Print Device Addresses (PRTDEVADR) command provides a printed configuration matrix of addresses for those twinaxial devices attached to a local work station controller. For each device attached to the local work station controller named in the controller description (CTLD parameter), the matrix shows the device's name, its port and switch setting, its type and model number, and whether the device is a display station or printer.

Required Parameters

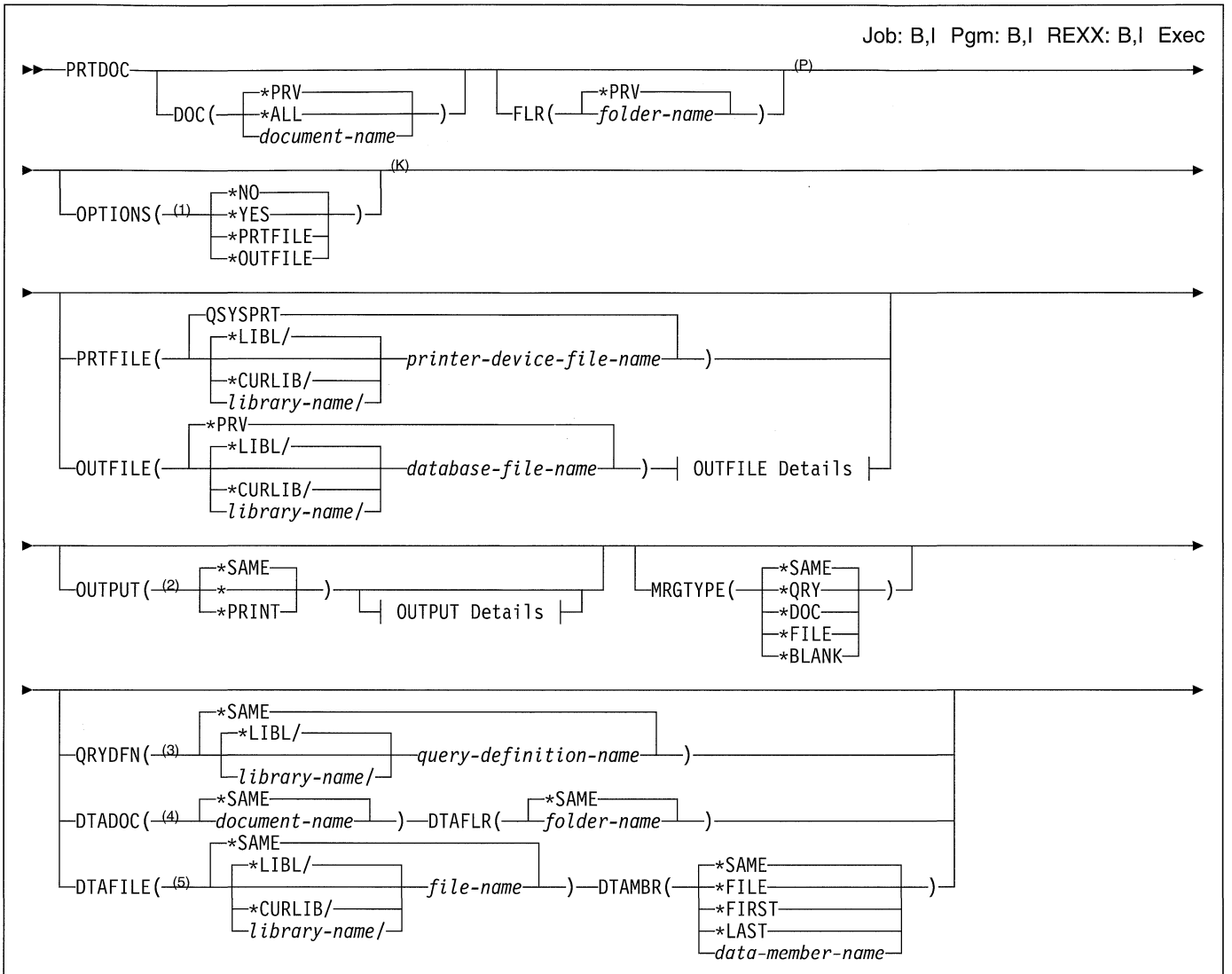
CTLD
Specifies the name of the local work station controller for which the matrix describes addresses of attached devices.

Example

```
PRTDEVADR  CTLD(CTL01)
```

This command prints a matrix of the devices that are attached to the CTL01 work station controller.

PRTDOC (Print Document) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 Batch cannot be used if OPTIONS(*YES) is specified.

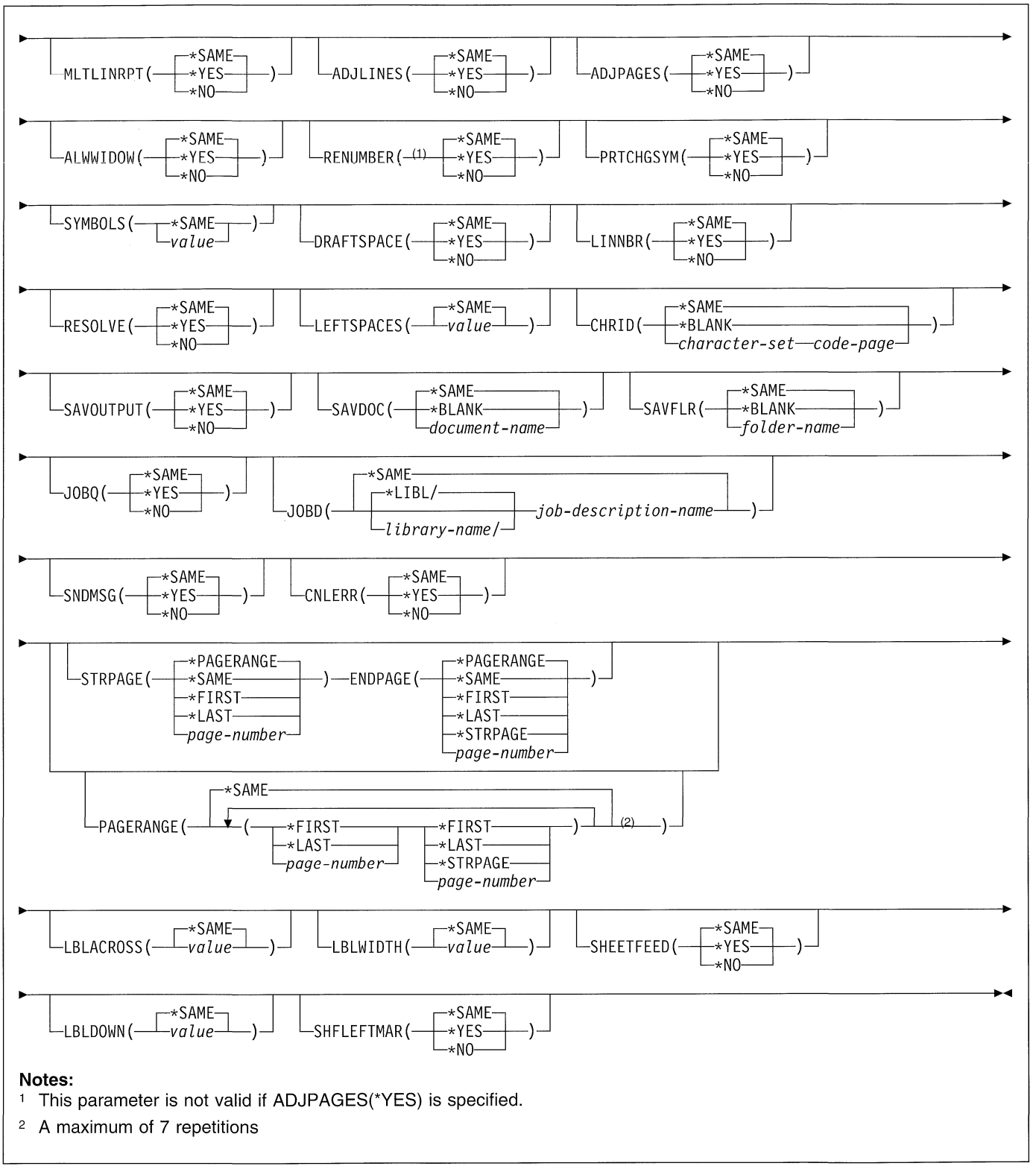
K All parameters preceding this point are key parameters.

2 Batch cannot be used if OUTPUT(*) is specified.

3 Only valid when MRGTYPE(*QRY) is specified.

4 Only valid when MRGTYPE(*DOC) is specified.

5 Only valid when MRGTYPE(*FILE) is specified.

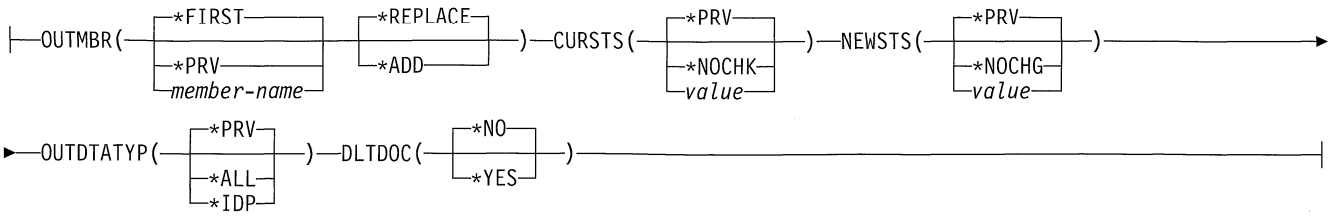


Notes:

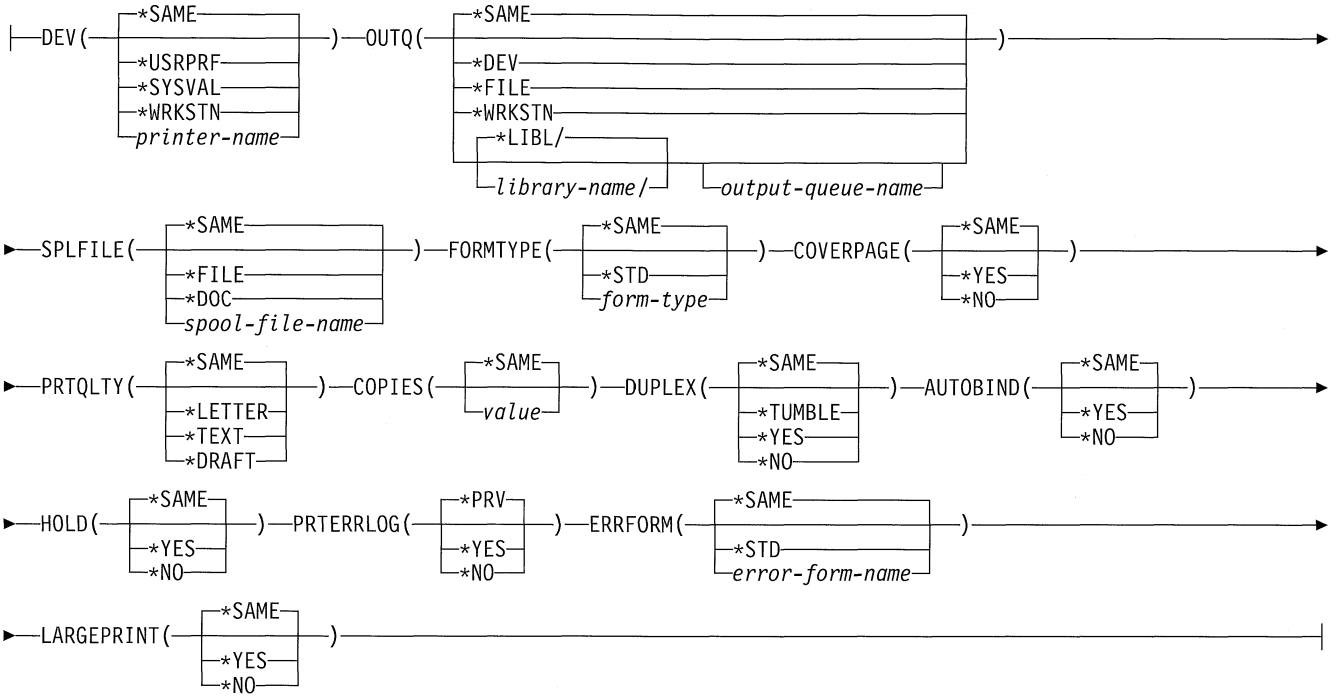
1 This parameter is not valid if ADJPAGES(*YES) is specified.

2 A maximum of 7 repetitions

OUTFILE Details:



OUTPUT Details:



Purpose

The Print Document (PRTDOC) command permits the user to print a document using the word processing function of OfficeVision/400.

This command also permits the user to override all print option values that are currently stored with a document. When a document is created, a set of default print options is associated with that document. If the user wants to override one or more of the parameters in this print command, the user must select OPTIONS(*YES) so that the print options appear on the display. When the print options appear, any of the print parameters can be changed. The user can override one or all of the print option parameters with this command. More information on printing documents is in the *Using OfficeVision/400* Word Processing*.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Optional Parameters

DOC

Specifies the name of the document to print.

***PRV:** The name used in the previous session is used.

***ALL:** All documents to which the user is authorized are printed to a database file. This is valid only when the output is directed to an OUTFILE.

document-name: Specify the name of the document that is printed.

FLR

Specifies the name of the folder that contains the document.

***PRV:** The name used in the previous session is used.

folder-name: Specify the name of the folder containing the document to be printed.

OPTIONS

Specifies whether the print options for this document are displayed before the document is printed.

***NO:** The print options are not displayed before the document is printed.

***YES:** The print options are displayed before the document is printed. Regularly used print options are set with this special value. For example, if STRPAGE(5) and OPTIONS(*YES) is specified, the value 5 appears on page 1 of the print options display.

***PRTFILE:** The print options specified in the PRTFILE parameter are used. When additional parameters are used, those parameters that are overridden by the appropriate printer file parameters when the document actually prints are not displayed.

Note: When additional parameters are used, those parameters not relevant to outfile processing are not displayed.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

PRTFILE

Specifies which printer file to use for the print options. The values found in the printer file for the print device, print quality, duplex, output queue, form type, copies, and hold override the corresponding values in the print options for the document. This parameter is valid only if OPTION(*PRTFILE) is also specified.

QSYSPRT: The document is printed using the system printer. This value overrides the printer name specified in the print options associated with the document.

The name of the printer device file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

printer-device-file-name: Specify the name of the printer file to use for this PRTDOC request. This value overrides the printer file name specified in the print options associated with the document.

OUTFILE

Specifies the qualified name of the database file where the displayed information is stored. If the specified file does not exist, this command creates a database file and file member. If the file is created, the text is labeled 'OUTFILE for PRTDOC' and the public authority is *EXCLUDE. Output to the database file is only supported if the OPTIONS (*OUTFILE) is also specified.

More information on defining the format of database files (OUTFILES) is in the *Office Services Concepts and Programmer's Guide*.

***PRV:** The library and database file used in the previous request is used.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the qualified name of the database file in which the resolved document information is stored.

OUTMBR

Specifies the name of the database file member to which the output is directed. If a member already exists, the system uses the second element of this parameter to determine whether the member is cleared before the new records are added. If the member does not exist and a member name is not specified, the system creates a member with the name of the output file specified on the OUTFILE parameter. If an output file member name is specified, but the member does not exist, the system creates it.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

***PRV:** The name used in the previous session is used.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

CURSTS

Specifies the value the document Interchange Document Profile (IDP) status field must have before the document may be printed to the database file. This field is 20 characters long and is valid only if OUTFILE output is requested.

Note: If the name of the document is specified in lowercase letters, the AS/400 system automatically shifts the name to uppercase letters. If the document name is to remain in lowercase letters, the name must be enclosed in apostrophes.

***PRV:** The name used in the previous session is used.

***NOCHK:** The status field is not checked before printing this document to a database file.

value: Specify the value to which the status field must be equal before the document is printed to the database.

NEWSTS

Specifies the value to which the document IDP status field value is set after the document has been printed to the database file. If a NEWSTS value is specified, the user must have at least *CHANGE authorization to the document. This field is 20 characters long and is valid only if OUTFILE output is requested.

***PRV:** The name used in the previous session is used.

***NOCHG:** The status field is not changed after printing this document to a database file.

value: Specify the value to which the status field is set after the document is printed to a database file.

OUTDTATYP

Specifies whether the entire document, or just the IDP information, is printed to the database file. This is valid only if OUTFILE output is requested.

***PRV:** The name used in the previous session is used.

***ALL:** The entire document is printed to a database file.

***IDP:** Only the IDP information is printed to a database file.

DLTDOC

Specifies whether a document is deleted after it has been printed to the database file. This is valid only if OUTFILE output is selected.

Note: The user must be the owner of the document or have *ALL authority to delete it.

***NO:** The document is not deleted after being printed to the database file.

***YES:** The document is deleted after being printed to the database file.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***SAME:** The output device specified in the document print options does not change.

***:** The document is shown on the display.

***PRINT:** The output is printed with the job's spooled output.

DEV

Specifies the name of the selected printer.

***SAME:** The name of the printer specified in the document print options does not change.

***USRPRF:** The printer ID indicated in the user profile is used to print the document.

***SYSVAL:** The value specified in the system value QPRTDEV is used.

***WRKSTN:** The printer assigned to the user's work station is used to print the document.

printer-name: Specify the name of the printer to use to print the document.

OUTQ

Specifies the qualified name of the output queue.

***SAME:** The output queue value specified in the document print options does not change.

***DEV:** The output queue specified on the PRTDEV parameter is used.

***FILE:** The output queue and output queue library values are based on:

1. If the PRTFILE parameter is specified, the values from the specified printer device are used.
2. If the PRTFILE parameter is not specified, the values from the printer file prompt on the document print options are used.

***WRKSTN:** The output queue assigned to the user's work station is used.

The name of the output queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

output-queue-name: Specify the name of the output queue to use to hold the output until it is ready to print.

SPLFILE

Specifies the name of the output file in which spooled files are kept.

***SAME:** The name of the output file specified in the document print options does not change.

***FILE:** The name chosen for the output file is the name for the printer file being used.

***DOC:** The document name is used for the spool file name. However, if the document name is longer than 10 characters or contains a period, the spool file name is QSYSPRT.

spool-file-name: Specify the name of the file to which to send the output. The file is then spooled to the queue.

FORMTYPE

Specifies the type of form on which the output is printed. The identifiers used to indicate the type of forms are user-defined and can be a maximum of 10 characters in length.

***SAME:** The type of form specified in the document print options does not change.

***STD:** The standard form type is used. The output is printed on the form type specified in the printer file for the printer selected. The printer file contains the information controlling how the document is printed on a particular printer.

form-type: Specify the type of form to use. Valid values range from 1 through 10 alphanumeric characters.

Note: Lowercase, blanks, or special characters must be enclosed in apostrophes. For example, a host system form type is entered as FORMTYPE(' '). The value is returned by the host system in a forms mount message.

COVERPAGE

Specifies whether a cover page is printed. The cover page includes such reference items as:

- Document name
- Folder name
- Document description
- Subject
- Reference, and
- Authors' names

***SAME:** The cover page value specified in the document print options does not change.

***YES:** The cover page is printed.

***NO:** The cover page is not printed.

PRTQLTY

Specifies the print quality used to print the document.

***SAME:** The print quality value specified in the document print options does not change.

***LETTER:** The letter quality (highest quality) is used.

***TEXT:** The text quality setting is used.

***DRAFT:** The draft quality (lowest quality) setting is used.

COPIES

Specifies the number of copies to print. This parameter only applies to spooled files.

***SAME:** The number of copies specified in the document print options does not change.

value: Specify a value ranging from 1 through 99 indicating the number of copies to print.

DUPLEX

Specifies whether output is printed on one side or two sides of the paper.

***SAME:** The duplex value specified in the document print options does not change.

***TUMBLE:** The document is printed on both sides of the paper. In addition, this special value indicates that the tops of both sides are on opposite ends of the page.

***YES:** The document is printed on both sides of the paper. In addition, this special value indicates that the tops of both sides are on the same end of the page.

***NO:** The document is printed on one side of the paper.

AUTOBIND

Specifies whether the left and right margins of the even pages will line up with the left and right margins of the odd pages when both sides of the paper are being printed.

***SAME:** The automatic binding value specified in the document print options does not change.

***YES:** The document is adjusted for binding.

***NO:** The document is not adjusted for binding.

HOLD

Specifies whether a print job is put on hold. Documents are held on the output queue and can be released to print or deleted. This parameter allows printing of several documents together by putting them on the output queue before releasing them to print.

***SAME:** The hold value specified in the document print options does not change.

***YES:** The print job is held.

***NO:** The print job is not held.

PRTERRLOG

Specifies whether a document error log is included as part of the information printed with the document.

***PRV:** The value used in the previous (last) PRTDOC request for this user is used.

***YES:** The error log is printed with the document.

***NO:** The error log is not printed with the document.

ERRFORM

Specifies the type of form on which to print the error log.

***SAME:** The error form value specified in the document print options does not change.

***STD:** The standard form type is used. The output is printed on the form type specified in the printer file for the selected printer. The printer file contains the information controlling how the document is printed on a particular printer.

error-form-name: Specify the name of the form on which to print the error log.

LARGEPRINT

Specifies whether the document is printed using large print.

***SAME:** The large print value specified in the document print options does not change.

***YES:** The document is printed in large print.

***NO:** The document is not printed in large print.

MRGTYPE

Specifies where data is located when it is merged.

***SAME:** The merge source specified in the document print options does not change.

***QRY:** The data is merged from a query. This query is a request to select and copy data from a file of one or more records based on the defined conditions.

***DOC:** The data stored in a document is merged.

***FILE:** The data stored in a file is merged.

***BLANK:** No data is merged.

QRYDFN
Specifies the name of the query that defines the data to be merged. This parameter is valid only when MRGTYPE(*QRY)is selected.

***SAME:** The query name specified in the document print options does not change.

The name of the query can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

query-definition-name: Specify the query name used to move the merged data.

DTADOC
Specifies the name of the document that contains the data being merged. This parameter is valid only when MRGTYPE(*DOC) is selected.

***SAME:** The document name specified in the document print options does not change.

document-name: Specify the name of the document by selecting a value ranging from 1 to 12 alphanumeric characters. If more than 8 characters are used, the ninth character must be a period (.) followed by a 1 to 3 character extension.

DTAFLR
Specifies the name of the folder that contains the document to merge. This parameter is valid only if MRGTYPE(*DOC) is selected.

***SAME:** The folder name specified in the document print options does not change.

folder-name: Specify the name of the folder that contains the document to merge.

DTAFILE
Specifies the file that contains the member to merge. This parameter is valid only when MRGTYPE(*FILE) is selected.

***SAME:** The file name specified in the document print options does not change.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the file that contains the data to merge.

DTAMBR
Specifies the name of the member that contains the data to merge. This parameter is valid only when MRGTYPE(*FILE) is selected.

***SAME:** The member name specified in the document print options does not change.

***FILE:** The member with the same name as the member name is used.

***FIRST:** The first member is used.

***LAST:** The last member is used.

data-member-name: Specify the name of the member that contains the data to merge.

MLTLINRPT
Specifies whether a multiple line report is created. In this report, data field instructions are merged to create records with several lines of output.

***SAME:** The multiple-line-report option specified in the document print options is used.

***YES:** A multiple line report is created.

***NO:** A multiple line report is not created.

ADJLINES
Specifies whether the line endings in a printed document are adjusted. The line endings are adjusted according to the specifications on the Line Spacing/Justification options display. This parameter is useful when printing a document that has been merged, has instructions, has display attributes that do not print spaces, or that uses a proportionally spaced font.

***SAME:** The line ending adjustment specified in the document print options does not change.

***YES:** The line endings are adjusted.

***NO:** The line endings are not adjusted. This special value is used when text is to be printed out in the same way that it was typed.

ADJPAGES
Specifies whether the page endings in a printed document are adjusted. The page ending adjustment is specified on the first typing line and last typing line prompts on the Page Layout/Paper Options display.

***SAME:** The page ending adjustment specified in the document print options does not change.

***YES:** The page endings are adjusted.

***NO:** The page endings are not adjusted.

ALWWIDOW

Specifies whether the document's page endings are determined by the exact number of lines per page specified on the Page Layout/Paper Options display.

| ***SAME:** The allow widow lines value specified in the
| document print options does not change.

***YES:** The page endings are determined by the exact number of lines per page.

***NO:** The page endings are not determined by the exact number of lines per page.

RENUMBER

Specifies whether the pages are renumbered when the document is printed.

***SAME:** The renumber system page numbers value specified in the document print options does not change.

***YES:** The page numbers are renumbered when the document is printed.

***NO:** The page numbers are not renumbered when the document is printed.

PRTCHGSYM

Specifies whether change symbols are printed in the left margin of the document. Change symbols are used to indicate all lines have been revised.

| ***SAME:** The print-change-symbol value specified in the
| document print options does not change.

***YES:** The change symbols are printed in the left margin of the document.

***NO:** The change symbols are not printed in the left margin of the document.

SYMBOLS

Specifies whether 5 different kinds of change symbols appear in the left margin of the document.

| ***SAME:** The change symbol value specified in the doc-
| ument print options does not change.

value: Specify up to 5 change symbol characters to appear in the left margin of the document.

DRAFTSPACE

Specifies whether the spacing value in the document can be adjusted. For example, if 3 (triple) is entered on the Line Spacing prompt, the double spacing value is 6, and 5 blank lines are printed between each line in the text of the document. Nevertheless, the document is still paginated using the value specified in the Line Spacing prompt. Therefore, depending on the amount of text being printed on a page, one page may print over onto a second page.

| ***SAME:** The draft space value specified in the docu-
| ment print options does not change.

***YES:** The space value in the document is doubled.

***NO:** The space value in the document is not doubled.

LINNBR

Specifies whether line numbers are printed in the document. The line numbers begin with 1 on the first page of the document. Line numbers are not printed in headers or footers.

| ***SAME:** The line numbers value specified in the docu-
| ment print options does not change.

***YES:** The line numbers are printed in the document.

***NO:** The line numbers are not printed in the document.

RESOLVE

Specifies whether the instructions placed in the document are processed. For example, the Date (.date) instruction is resolved to the actual date, which, in this example, is 04/03/62.

| ***SAME:** The resolve value specified in the document
| print options does not change.

***YES:** The instructions placed in the document are resolved.

***NO:** The instructions placed in the document are not resolved. For example, the Date instruction (.date) prints as ***date**.

LEFTSPACES

Specifies whether the left margin of the document is increased.

| ***SAME:** The left-spaces value specified in the docu-
| ment print options does not change.

value: Specify a value, ranging from 0 through 99, for the number of spaces to add to the left margin in the printed document.

CHRID

Specifies the character identifier (graphic character set and code page) for the file. This parameter allows printing of text that is in different character identifier (graphic character set and code page) coding. The value specified on this parameter is used to instruct the printer device to interpret the hexadecimal byte string to print the same characters that were intended when the text was created. More information about the character identifier is in the *Guide to Programming for Printing*. A list of valid CHRID values and applicable printers is in
| the "CHRID Values and Applicable Printers (CHRID
| parameter)" table in Appendix B, "Font, Character Identi-
| fier, and Other Values Supported for Different Printers."

| ***SAME:** The graphic character set id specified in the
| document print options does not change.

***BLANK:** Text is not specified.

Element 1: Character Set

character-set: Specify the type of graphic character set to use by entering the appropriate 3-digit identifier.

Element 2: Code Page

code-page: Specify the code page value used to create the command parameters. Valid values range from 1 through 999.

SAVOUTPUT
Specifies whether the document being printed is also saved as a final form document.

***SAME:** The save resolved output value specified in the document print options does not change.

***YES:** The printed document is saved as a final form document.

***NO:** The printed document is not saved as a final form document.

SAVDOC
Specifies the name of the document that contains the final form document.

***SAME:** The save document name specified in the document print options does not change.

***BLANK:** A resolved output document is not specified.

document-name: Specify the name of the document that contains the resolved document. The document name must range from 1 to 12 characters in length. If more than 8 characters are selected, the ninth character must be a period (.) followed by a 1 to 3 character extension. If the document name specified does not exist, a document is created.

SAVFLR
Specifies the name of the folder that contains the document being saved in final form.

***SAME:** The save folder value specified in the document print options does not change.

***BLANK:** A resolved output folder is not specified.

folder-name: Specify the name of the folder to contain the final-form document.

JOBQ
Specifies whether the print request is put on the job queue.

***SAME:** The place on the job queue that is specified in the document print options does not change.

***YES:** The printing of the document is placed in the job queue.

***NO:** The printing of the document is not placed in the job queue.

JOBD
Specifies the name of the job description that describes how the printing job is run.

***SAME:** The place on the job queue that is specified in the document print options does not change.

The name of the job description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-description-name: Specify the name of the job description.

SNDMSG
Specifies whether a print job has been sent to the job queue and the user wants to receive a message specifying that the job has completed.

***SAME:** The message value specified in the document print options does not change.

***YES:** A message is sent to the user when the print job has completed.

***NO:** A message is not sent to the user when the print job has completed.

CNLERR
Specifies whether printing is stopped if an error is detected within the document.

***SAME:** The cancel error value specified in the document print options does not change.

***YES:** Printing is stopped on the document if an error is detected. An error message stating that the job is canceled is listed in the error log.

***NO:** Printing does not stop if an error is detected.

STRPAGE
Specifies the page on which printing begins.

Note: If the STRPAGE(*page-number*) value specified is larger than the ENDPAGE(*page-number*) value specified, the entire document is printed.

***PAGERANGE:** The pages specified on the PAGERANGE parameter are printed.

***SAME:** The start page specified in the document print options does not change.

Note: If STRPAGE(*SAME) is specified and additional page ranges exist in the document print options, an error message is sent and the document is not printed.

***FIRST:** Printing is started on the first page of the document.

***LAST:** Printing is started on the last page of the document.

page-number: Specify the page on which to begin printing. Valid values range from 0.01 through 9999.99.

ENDPAGE
Specifies the page on which printing ends.

***PAGERANGE:** The pages specified on the PAGERANGE parameter are printed.

***SAME:** The end page value specified in the document print options does not change.

Note: If ENDPAGE(*SAME) is specified and additional page ranges exist in the document print options, an error message is sent and the document is not printed.

***FIRST:** Printing is ended after the first page of the document.

***LAST:** Printing is ended after the last page of the document.

***STRPAGE:** The end page value is the same as the start page value. Only one page is printed.

page-number: Specify the page on which to stop printing. Valid values range from 0.01 through 9999.99.

PAGERANGE
Specifies the page ranges to print. A maximum of 7 ranges can be specified.

***SAME:** The page range specified on the document print options is printed.

Element 1: Start Page

***FIRST:** Printing is started on the first page of the document.

***LAST:** Printing is started on the last page of the document.

page-number: Specify the page on which to begin printing. Valid values range from 0.01 through 9999.99.

Element 2: End Page

***FIRST:** Printing is ended after the first page of the document.

***LAST:** Printing is ended after the last page of the document.

***STRPAGE:** The end page value is the same as the start page value. Only one page is printed.

page-number: Specify the page on which to stop printing. Valid values range from 0.01 through 9999.99.

LBLACROSS
Specifies the number of labels to print across the page.

***SAME:** The label-across-the-page value specified in the document print options does not change.

value: Specify a value, ranging from 1 through 99, that indicates the number of labels to print across the page.

LBLWIDTH
Specifies how wide to make the label. The width of a label is the number of characters from the left edge of the first label to the left edge of the next label, including the blank spaces between the labels. If the width specified is larger than the margins specified for the document, the margins are used as the width.

***SAME:** The label width value specified in the document print options does not change.

value: Specify a value, ranging from 2 through 198, that indicates the label width.

SHEETFEED
Specifies whether sheet feed paper is used for printing and whether there are more than one row of labels on a page. When using sheet feed paper, this is the only parameter to use to print more than one row of labels on a page.

***SAME:** The sheet feed value specified in the document print options does not change.

***YES:** Sheet feed printing is used and there are more than one row of labels on a page.

***NO:** Sheet feed printing is not used, or there is only one row of labels on a page.

LBLDOWN
Specifies the number of rows of labels to print on a page.

***SAME:** The label down value specified in the document print options does not change.

value: Specify a value, ranging from 1 through 99, that indicates the number of rows of labels to be printed on a page.

SHFLEFTMAR
Specifies whether to shift the left margin to prevent text from being truncated.

***SAME:** The SHFLEFTMAR value does not change.

***YES:** When the right margin exceeds the paper edge, the left margin is shifted so that as much text as possible is printed. If the right margin does not exceed the paper edge, the text is not shifted.

***NO:** The left margin is not shifted when text exceeds the right margin. Any text exceeding the right margin is truncated.

Examples

Example 1: Printing to a File

```
PRTDOC  DOC(MYDOC)  FLR(MYFLR)  OPTIONS(*OUTFILE)
  OUTFILE(MYFILE/MYLIB)  OUTMBR(MYMBR *REPLACE)
  CURSTS(*PRV)  NEWSTS(*PRV)  OUTDTATYP(*PRV)
  PRTERRLOG(*PRV)  DLTDOC(*NO)
```

This command prints the document MYDOC in folder MYFLR to the database file MYFILE in library MYLIB in the database file member MYMBR. If the member already exists, it is replaced by the contents of MYDOC. The CURSTS, NEWSTS, OUTDTATYP, and PRTERRLOG are taken from the last PRTDOC request. The document is not deleted after it is printed to the database file MYFILE.

Example 2: Printing a Document

```
PRTDOC  DOC(MYDOC)  FLR(MYFLR)  OPTIONS(*NO)
  DEV(MYPRNTR)  OUTQ(*DEV)
```

This command prints the document MYDOC in the folder MYFLR on a printer called MYPRNTR.

Example 3: Printing Document Error Log

```
PRTDOC  DOC(MYDOC)  FLR(MYFLR)  OPTIONS(*NO)
  PRTERRLOG(*YES)
```

This command prints the document with a document error log attached to it.

Example 4: Increasing Margin

```
PRTDOC  DOC(MYDOC)  FLR(MYFLR)  OPTIONS(*NO)
  LEFTSPACES(10)
```

This command prints the document and has 10 extra spaces inserted in the left margin.

Example 5: Printing a Cover Page

```
PRTDOC  DOC(MYDOC)  FLR(MYFLR)  OPTIONS(*NO)
  COVERPAGE(*YES)
```

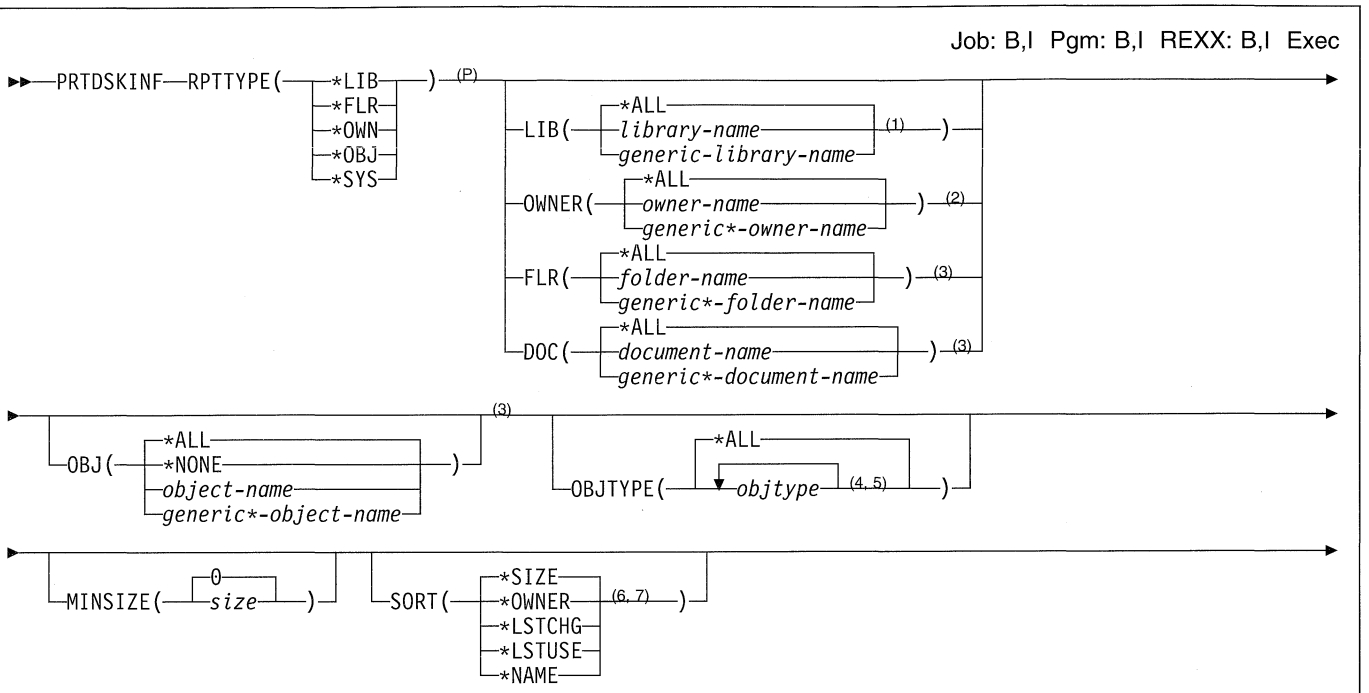
This command prints the document with a cover page.

Example 6: Printing One Page to a File

```
PRTDOC  DOC(MYDOC)  FLR(MYFLR)  OPTIONS(*OUTFILE)
  OUTFILE(MYLIB/MYFILE)
  OUTMBR(*FIRST)  PAGERANGE((5 5))
```

This command prints page 5 of the document to the database file MYFILE in library MYLIB in the first member.

PRTDSKINF (Print Disk Information) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 This parameter is valid only if *LIB is specified on the RPTTYPE parameter.

2 This parameter is valid only if *OWN is specified on the RPTTYPE parameter.

3 This parameter is valid only if *FLR is specified on the RPTTYPE parameter.

4 A maximum of 60 repititions

5 A list of the valid OS/400 object types for this command is in Appendix A.

6 This parameter is not valid if *OWN is specified on the RPTTYPE parameter, unless the OBJ parameter is used with a value other than *NONE.

7 This parameter is not valid if *SYS is specified on the RPTTYPE parameter.

Purpose

The Print Disk Information (PRTDSKINF) command is used to print disk space information that is already stored in the database file QAEZDISK. The output with file name QPEZDISK goes to the spool queue associated with the job using this command.

Required Parameters

RPTTYPE

Specifies the type of report to print. The report information is taken from member QCURRENT in QAEZDISK. If QCURRENT does not contain any data, an error message is sent.

***LIB:** A report of the library information contained in the file is printed.

***FLR:** A report of the folder information contained in the file is printed.

***OWN:** A report of the user profile (owner) information contained in the file is printed.

***OBJ:** A report of object information contained in the file is printed.

***SYS:** A report of only the system information contained in the file is printed.

Optional Parameters

LIB

Specifies the names of the libraries to print information about.

***ALL:** The report has information on all user libraries on the system.

library-name: Specify the user library.

generic-library-name:* Specify the generic library name. A generic name is a character string of one or more characters followed by an asterisk (*); for example,

ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be printed only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

OWNER

Specifies the names of the owners (user profiles) to print information about.

***ALL:** The report contains information on all user profiles on the system.

owner-name: Specify the user profile.

generic-owner-name:* Specify the generic user profile.

FLR

Specifies the names of the folders to print information about.

***ALL:** The report has information on all user folders on the system.

folder-name: Specify the folder name.

generic-folder-name:* Specify the generic folder name.

DOC

Specifies the names of the documents to print information about.

***ALL:** The report contains information on all documents in the specified folder.

document-name: Specify the document by the given name within the specified folder.

generic-document-name:* Specify the documents specified by the generic qualification.

OBJ

Specifies the names of the objects to print information about.

***ALL:** If you specify a library or owner, then the object information is all objects within the library or those controlled by the owner

***NONE:** No library or owner is specified.

object-name: Specify a library or owner, then the object information is the object specified by the given name within the library or controlled by the owner.

generic-object-name:* Specify a library or owner, then the object information are the objects that meet the specified generic qualification within the library or controlled by the owner.

OBJTYPE

Specifies the object types to print information about.

***ALL:** If you specify a library or owner, information is printed on all the specified object types within the library or controlled by the owner. If an object name is specified, information on all object types within that name, within the library, or controlled by the owner is printed. If a library or owner is not specified, the report has information on all object types on the system. If an object name is specified, information only on object types with that name is printed.

object-type: Specify a library or owner, then the object type information is the object type specified within the library or controlled by the owner. If an object is specified, the report has information on the objects with the specified object type within the library or controlled by the owner.

MINSIZE

Specifies the size of the smallest piece of information to include. For example, if a library report is requested without objects, then this size would be the size of the smallest library to include. If objects within the library are requested, then this would be the size of the smallest object within the library to include.

0: All objects are included regardless of size.

size: Specify size in thousands of bytes.

SORT

Specifies the order in which the information should be sorted.

***SIZE:** Information is sorted from large to small.

***OWNER:** The information is sorted in alphabetical order by owner name.

***LSTCHG:** The information is sorted by last-change date with the oldest information first.

***LSTUSE:** The information is sorted by last-use date with the oldest information first.

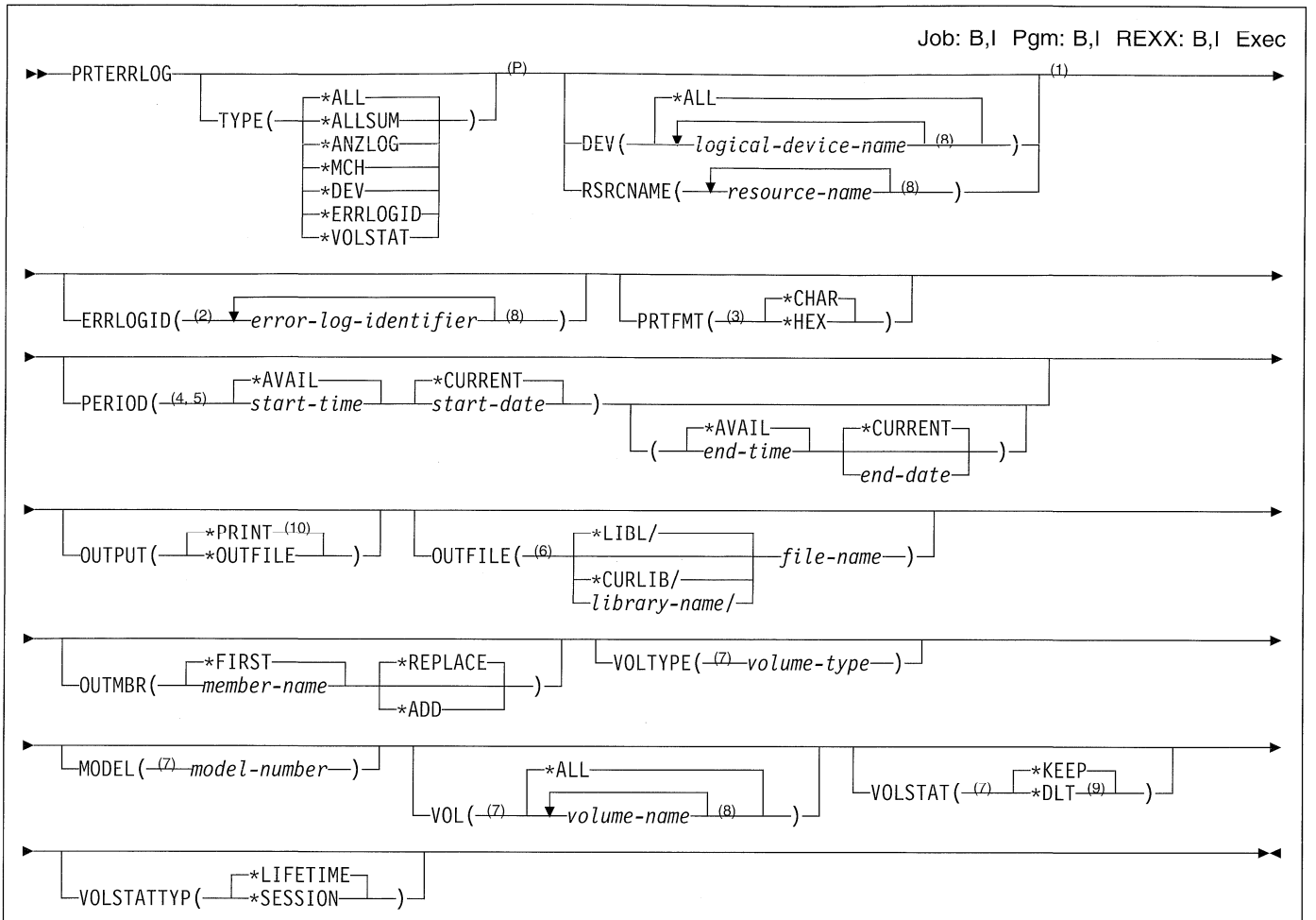
***NAME:** Information is sorted in alphabetical order according to the report type.

Example

```
PRTDSKINF   RPTTYPE(*LIB)  LIB(*ALL)  OBJ(*ALL)
  SORT(*SIZE)
```

This command prints a library report from database file QAEZDISK in library QUSRSYS in member QCURRENT, containing information about all libraries, objects, and object types in the libraries. The information is sorted by size and sent to the printer file QPEZDISK.

PRTERRLOG (Print Error Log) Command



Notes:

1 DEV and RSRCNAME are valid only if TYPE(*DEV) is specified.

2 ERRLOGID is valid only if TYPE(*ERRLOGID) is specified.

3 PRTFMT is not valid if TYPE(*VOLSTAT), TYPE(*ALLSUM), or OUTPUT(*OUTFILE) is specified.

4 PERIOD contains 2 lists of 2 elements each. *N must be specified for any element that goes before the value specified. PERIOD not valid when TYPE(*ERRLOGID) is specified.

5 PERIOD is not valid when TYPE(*VOLSTAT) and VOLSTATTYP(*LIFETIME) are specified.

6 OUTFILE is valid only with TYPE(*DEV), TYPE(*ERRLOGID), or TYPE(*VOLSTAT).

7 VOLTYPE, MODEL, VOL, VOLSTAT, and VOLSTATTYP are valid only if TYPE(*VOLSTAT) is specified.

8 A maximum of 10 repetitions

9 *DLT is not valid if OUTPUT(*OUTFILE) is specified.

10 *PRINT is not valid if VOLSTATTYP(*SESSION) is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Print Error Log (PRTERRLOG) command is used primarily for problem analysis. It places a formatted printer file of the data in the machine error log in a spooled printer device file named QPCSMPRT or in a specified output file. The error log data can be used by the IBM service representatives.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Optional Parameters

TYPE

Specifies the type of error log data from the machine error log to print in the spooled printer file.

***ALL:** All the error codes in the machine's error log are printed. In addition, the error codes for each subsystem (for example, direct access storage devices or printers) are printed in summary form.

***ALLSUM:** All the data in the error log is printed in summary form.

***ANZLOG:** A one-line summary is created for each entry in the error log.

Note: All other parameters except PERIOD are ignored when TYPE(*ANZLOG) is specified.

***MCH:** Only the error data produced by machine checks is printed.

***DEV:** Only the error data produced by the devices specified on the DEV and RSRCNAME parameters are printed.

***ERRLOGID:** Only the error data with the specified error log record is printed. If this value is specified, the ERRLOGID parameter must also be specified. It is ignored for other request types.

***VOLSTAT:** If this value is specified and if OUTPUT(*PRINT) is specified, the tape or diskette volume 'lifetime' statistical data records are printed. If this value is specified and if OUTPUT(*OUTFILE) is specified, 'session' volume statistical data records are printed.

DEV

Specifies the device names for which the user wants the error log data printed.

Note: This parameter is valid only if TYPE(*DEV) is specified. This parameter cannot be specified if the RSRCNAME parameter is specified.

***ALL:** The error log data is printed for all device names.

logical-device-name: Specify one or more device names for which error log data is to be printed. Up to ten device names can be specified.

RSRCNAME

Specifies that those error log entries with device information based on the specified resource name are printed. This parameter cannot be specified if the DEV parameter is specified. This parameter is valid only if TYPE(*DEV) is specified. Up to ten resource names can be specified.

ERRLOGID

Specifies that error log entries with the specified error log identifier are printed. This parameter is valid only if TYPE(*ERRLOGID) is specified. It is ignored for other request types. Up to ten error log identifiers can be specified.

PRTFMT

Specifies that the indicated report prints any hexadecimal data in character format. This parameter cannot be specified if TYPE(*VOLSTAT) is specified or if the OUTFILE parameter is specified.

***CHAR:** The report is formatted so that hexadecimal data prints as character data.

***HEX:** The report is formatted so that hexadecimal data is printed in hexadecimal format.

Note: Specifying PRTFMT(*HEX) can cause spooling or printing of large amounts of data. This can impact overall system performance.

PERIOD

Specifies the period of time for which the error log data is printed. This parameter is specified as two sets of two values each.

Element 1: Start Time

***AVAIL:** The error data that is available for the specified starting date is printed.

start-time: Specify the start time on the specified start date for which the error data is printed. The time is specified in 24-hour format with or without a time separator as follows:

- With a time separator, specify a string of 5 or 8 digits where the time separator separates the hours, minutes, and seconds. If this command is entered from the command line, the string must be enclosed in apostrophes. If a time separator other than the separator specified for your job is used, this command fails.
- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where **hh** = hours, **mm** = minutes, and **ss** = seconds. Valid values for **hh** range from 00 through 23. Valid values for **mm** and **ss** range from 00 through 59.

Element 2: Start Date

***CURRENT:** The error data that is available for the current day and between the specified start and end times (if specified) is printed.

start-date: Specify the starting date. The date must be specified in the job date format.

Element 3: End Time

***AVAIL:** The error data that is available for the specified end date is printed.

end-time: Specify the ending time for the specified ending date that determines the error data that is printed.

Element 4: End Date

***CURRENT:** The error data that is available for the current day and between the specified start and end times (if specified) is printed.

end-date: Specify the end date for which error data is printed. The date must be specified in job date format.

OUTPUT
Specifies whether the output from the command is printed with the job's spooled output or directed to a database file. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

OUTFILE
Specifies the database file to which the report is directed. If the output file already exists, the system attempts to use it. Records replace or are added to existing data in the file member. If the output file does not exist, the system creates a database file in the specified library. A member is created for the file with the name specified in the OUTMBR parameter.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the file that will receive the report.

OUTMBR
Specifies the name of the database file member to which the output is directed. If a member already exists, the system uses the second element of this parameter to determine whether the member is cleared before the new records are added. If the member does not exist and a member name is not specified, the system creates a member with the name of the output file specified on the OUTFILE parameter. If an output file member name is specified, but the member does not exist, the system creates it.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

VOLTYPE
Specifies the volume type of the specified volume identifier. Valid values are 4-digit device type numbers for cartridge tape, reel tape, or diskette.

MODEL
Specifies the model number of the specified model type. This parameter is required if TYPE(*VOLSTAT) is specified.

VOL
Specifies one or more volume identifiers used by the file. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ALL:** Volume statistics are processed for all volumes.

volume-name: Specify the name of the volume for which statistics are processed. A maximum of 10 volume names can be specified.

VOLSTAT
Specifies whether the volume statistical data records are kept or deleted from the machine error log after they are printed. This parameter is valid only if TYPE(*VOLSTAT) is specified.

Note: ENDOPT(*UNLOAD) must be specified during the SAVE operation to generate volume statistics at the completion of the tape operation.

***KEEP:** The volume statistical data records are kept in the error log after they are printed.

***DLT:** The volume statistical data records are deleted from the error log for volumes that are not active after they are printed.

Note: If OUTPUT(*OUTFILE) is specified, *DLT cannot be specified.

VOLSTATTYP
Specifies the type of volume statistics printed or directed to an output file. This parameter is valid only if TYPE(*VOLSTAT) is specified.

***LIFETIME:** Lifetime statistics are printed. Lifetime statistics cannot be placed in an output file.

***SESSION:** Session statistics are directed to the output file specified on the OUTFILE parameter. Session statistics cannot be printed.

Examples

Example 1: Printing Error Log Data

```
PRTERRLOG
```

This command gets the error data in the machine error log that occurred for all device types and puts it in a spooled file. The entire error log is printed and any hexadecimal data is in character format.

Example 2: Using the System Resource Manager Database

```
PRTERRLOG  TYPE(*DEV)  RSRCNAME(TAPE000001)
  PRTFMT(*HEX)
```

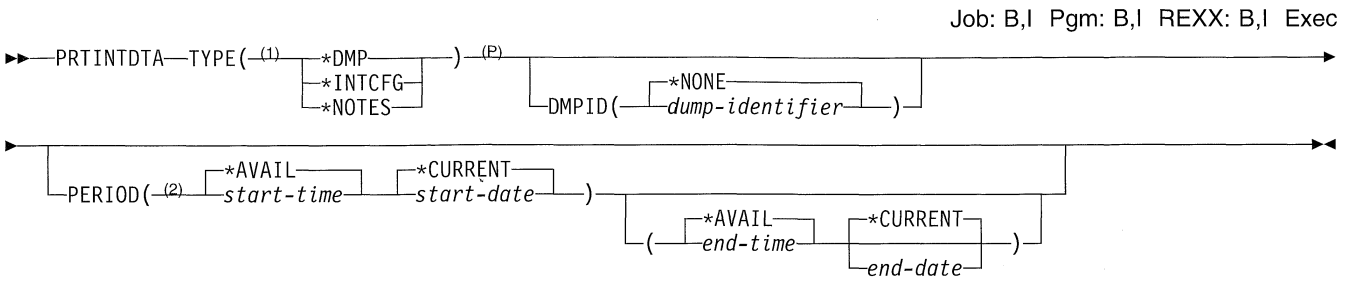
This command uses the system resource manager database to determine the device type, model, and serial number for the resource TAPE000001. The print request is based on that information. The report is put in the spooled file and contains all records that pertain to that device type, model, and serial number. Any hexadecimal data in the file is converted to hexadecimal format.

Example 3: Processing Error Log Entries

```
PRTERRLOG   TYPE(*DEV)  DEV(DISKLUD1)
  OUTPUT(*OUTFILE)  OUTFILE(MYLIB/MYDBD)
  OUTMBR(ELOG)
```

This command processes all the error log entries for the logical device named DISKLUD1. They are put in the file MYDBD, in the library MYLIB, and in the member ELOG. No spooled files are created.

PRTINTDTA (Print Internal Data) Command



Notes:

1 If *DMP is specified, the DMPID parameter must be specified.

P All parameters preceding this point can be specified in positional form.

2 PERIOD contains two lists of two elements each. PERIOD values must be coded in this order, using *N for those not being specified: (start-time start-date) (end-time end-date). PERIOD is valid only if TYPE(*NOTES) is specified.

Purpose

The Print Internal Data (PRTINTDTA) command is used primarily for problem analysis. It writes the machine internal data to a spooled printer file. The data is used to service the system. The names of files produced by this and other commands is in the *Programming Reference Summary*.

Restrictions:

1. This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.
2. This command is intended for use by service representatives only.

Required Parameters

TYPE

Specifies the type of data to be printed.

***DMP:** The data to be printed was dumped by a previously issued Dump Job Internal (DMPJOBINT) command or by the machine when it processes a device error or object damage. The dump identifier of the data must be specified on the DMPID parameter.

***INTCFG:** The machine internal configuration and resource information is printed.

***NOTES:** The notes portion of the machine internal data, for the period specified by the PERIOD parameter, is printed.

Optional Parameters

DMPID

Specifies, for internal dump operations only, the dump identifier associated with the machine internal data that is printed. This parameter must be specified *only* if TYPE(*DMP) is specified; otherwise, it is ignored. The dump identifier is sent in one of three ways:

- In a message to the job that issued the DMPJOBINT command that dumped the internal data
- In a damage message
- In a device error message or machine function check message. The message containing the dump identifier may also be sent to the system history log.

***NONE:** No dump identifier is specified.

dump-identifier: Specify the dump identifier of the dump output that is printed. The identifier specified must contain eight characters.

PERIOD

Specifies the period of time for which the notes portion of the machine internal data is printed. This parameter is valid only if TYPE(*NOTES) is specified; otherwise, it is ignored. The following values can be coded in this parameter, which contains two sets of two values each. Refer to the syntax diagram for the location of each value specified. If this parameter is not specified, all the available notes for the current date are printed.

Element 1: Starting Time

***AVAIL:** The notes that are available from the starting date to the ending date (or for the current day only) are printed.

start-time: Specify the starting time for the specified starting date for which the user wants the notes printed. The time is specified in 24-hour format with or without a time separator as follows:

- With a time separator, specify a string of 5 or 8 digits where the time separator separates the hours, minutes, and seconds. If this command is entered from the command line, the string must be enclosed in apostrophes. If a time separator other than the

separator specified for your job is used, this command fails.

- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where **hh** = hours, **mm** = minutes, and **ss** = seconds. Valid values for **hh** range from 00 through 23. Valid values for **mm** and **ss** range from 00 through 59.

Element 2: Starting Date

***CURRENT:** The notes that are available for the current day and between the specified starting and ending times (if specified) are printed.

start-date: Specify the date printed. The date must be entered in the format specified by the system values QDATFMT and, if separators are used, QDATSEP.

Element 3: Ending Time

***AVAIL:** The notes that are available from the starting date to the ending date (or for the current day only) are printed.

end-time: Specify the ending time for the specified ending date that determines the notes that is printed.

Element 4: Ending Date

***CURRENT:** The notes that are available for the current day and between the specified starting and ending times (if specified) are printed.

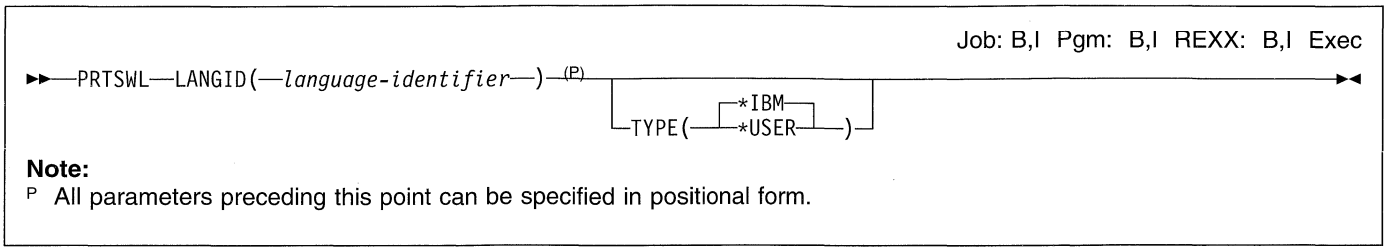
end-date: Specify the ending date after which the notes are no longer printed. The system date format must be used.

Example

```
PRTINTDTA  TYPE(*DMP)  DMPID(0102FA3C)
```

This command prints the job internal dump output that has a dump identifier of 0102FA3C.

PRTSWL (Print Stop Word List) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Print Stop Word List (PRTSWL) command is used to print the words from an IBM-supplied or user-created stop word list.

Required Parameters

LANGID

Specifies the language identifier (ID) for the stop word list.

Optional Parameters

TYPE

Specifies the type of stop word list to print.

***IBM:** The stop word list is IBM-supplied.

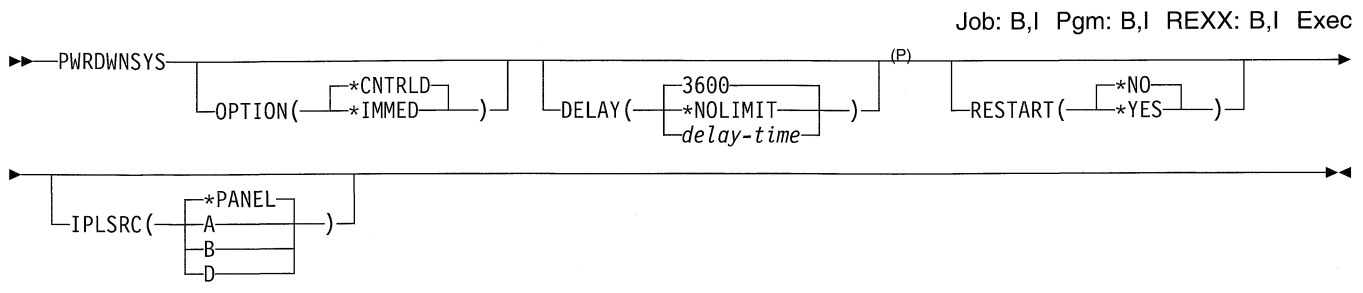
***USER:** The stop word list is user-created.

Example

```
PRTSWL  LANGID(ENG)  TYPE(*IBM)
```

This command prints the IBM-supplied stop word list with the language ID ENG.

PWRDWNSYS (Power Down System) Command



Note:

(P) All parameters preceding this point can be specified in positional form.

Purpose

The Power Down System (PWRDWNSYS) command prepares the system for ending and then starts the power-down sequence. All active subsystems are notified that the system is being powered down; no new jobs or routing steps can be started by any subsystem. For example, jobs that are on a job queue as a result of a Transfer Job (TFRJOB) command are not allowed to complete. During the subsequent initial program load (IPL), they are removed from the job queue and their job logs are produced.

Note: If tape units are installed on the system, all tape reels that are on the devices should be unloaded *before* the system is powered down. This step ensures the integrity of data on the tapes.

Restriction: To run this command, the user must have job control (*JOBCTL) authority.

Optional Parameters

OPTION

Specifies whether the system either allows the active subsystem to end processing of active jobs in a controlled manner (which lets the application program perform end processing) or the system ends the jobs immediately. In either case, the system performs certain job-cleanup functions.

***CNTRLD:** The subsystem, within the time specified by the DELAY parameter, ends all active jobs in a controlled manner. During that time, programs running in those jobs are allowed to perform job cleanup (end-of-job processing) functions. If an active job could begin to loop or send an inquiry message to QSYSOPR, an appropriate time delay should be specified by using the DELAY parameter.

***IMMED:** The subsystem ends all active jobs immediately. This means the programs running in those jobs are not allowed to perform any job cleanup. Thus, a minimum amount of time is required when *IMMED is specified. The amount of time allowed for cleanup when *IMMED is specified is controlled by QPWRDWNLMT, the system value. This option might cause undesirable results if data has been partially updated, and it should be used only after a controlled end has been unsuccessfully attempted.

When OPTION(*IMMED) is specified while the system is operating under auxiliary power, or if the delay time specified in the DELAY parameter ends while the system is under auxiliary power, the system ignores the QPWRDWNLMT system value and starts the power-down sequence without additional job cleanup activity.

DELAY

Specifies the number of seconds that the system allows a controlled end of processing operation to be performed by the active subsystems. If the end-of-job cleanup functions are not finished within the specified delay time, any remaining jobs are ended immediately.

3600: The amount of time in which to complete a controlled end of processing is limited to 3600 seconds.

***NOLIMIT:** The system does not power down until the last job is complete.

Note: If *NOLIMIT is specified and a batch job begins to loop, the system does not power down.

delay-time: Specify the maximum number of seconds in which a controlled end operation can be performed. Valid values range from 1 through 99999 seconds.

RESTART

Specifies whether the system ends and powers down, or ends and then restarts in unattended mode.

***NO:** The system ends and powers down.

***YES:** If the system is on utility power, it undergoes end of system processing (but does not power down) and then does an initial program load (IPL). If the system is on auxiliary power, it powers down and does an automatic IPL when utility power is restored (if the QPWRRSTIPL system value is set to '1'). When the system restarts or an automatic IPL occurs, the IPL proceeds in an unattended mode. In unattended mode, dis-

plays such as the *IPL options* prompt are not shown. More information is in the *Work Management Guide*.

IPLSRC

Specifies whether an initial program load (IPL) is started from the A-source, B-source, or D-source of the system. This parameter allows the user to control which Vertical Licensed Internal Code (VLIC) storage source of the system to use at IPL. Also, specifying the source of the system helps the user to determine the appropriate source in which to send VLIC during program temporary fixes (PTFs).

Source Considerations:

VLIC has three storage areas, known as the A-source, the B-source, and the D-source. The D-source is a tape device. The A- and B-sources are part of the system memory. Initially, the A- and B-sources are identical, but when Licensed Internal Code fixes are performed temporarily (PTF), the temporary fixes are stored only on the B-source. When these fixes become permanent, they are copied from the B-source to the A-source; therefore, the fixes reside on both the A-source and B-source.

The user who wants to send temporary fixes to the B-source must start the system from the A-source, which causes the fixes to be sent to the opposite source, or the B-source.

A user that starts the system from the A-source is running the system from the permanent fixes. A user that starts the system from the B-source is running the system from a combination of temporary and permanent fixes. A user that starts the system from the D-source uses the Licensed Internal Code loaded from the media in the tape device.

Notes:

1. If the system is not running on Version 2, Release 1.1 or a subsequent release, specifying the D-source will fail.
2. It is recommended that the user specify RESTART(*YES), otherwise, the user cannot be assured as to which source the system is actually starting. This precaution can save the user time.

***PANEL:** The system is started from the source (A-source, B-source, or D-source) that is currently shown on the operator's display,

A: The system is started from the A-source.

B: The system is started from the B-source.

D: The system is started from the D-source, a tape device.

Examples

Example 1: Performing An Immediate End

```
PWRDWNSYS  OPTION(*IMMED)
```

This command causes the system to perform an immediate end without allowing any active jobs to perform cleanup routines. Once the system completes its end functions, it starts the power-down sequence.

Example 2: Specifying a Controlled End

```
SBMJOB  JOB(LASTJOB)  JOBD(QBATCH)  JOBPTY(9)
  JOBQ(QBATCH)  RQSDTA('PWRDWNSYS *CNTRLD 3600')
```

This command submits a low priority batch job that, when run, causes the system to perform a controlled end. The controlled end is allowed one hour (3600 seconds) for completion before any remaining jobs are ended. This method of issuing the PWRDWNSYS command could be used to allow other higher priority jobs on job queue QBATCH (including those that are on the queue as a result of the Transfer Job (TFRJOB) command) to be completed before the PWRDWNSYS command is run. There must be an active subsystem for which the QBATCH job queue is a source of work.

Example 3: Specifying a Controlled End With No Time Limit

```
PWRDWNSYS  OPTION(*CNTRLD)  RESTART(*YES)
```

This command causes the system to perform a controlled end with no time limit. When all jobs in the system have completed, the system prepares for ending and starts an IPL.

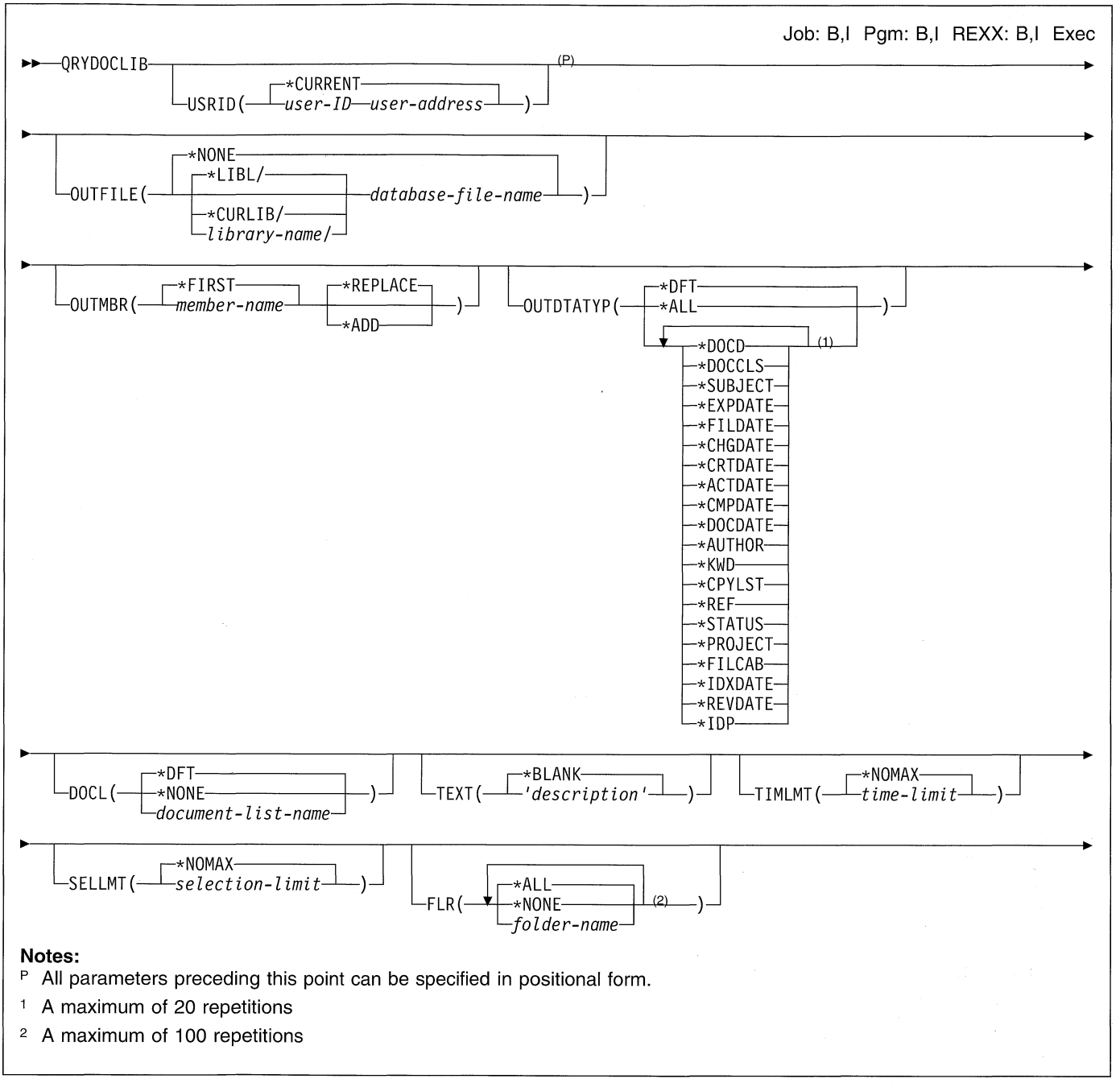
After PWRDWNSYS OPTION(*CNTRLD) is entered, and before the delay time ends, this command can be overridden by entering PWRDWNSYS OPTION(*IMMED). In this case, the values specified or defaulted for the RESTART parameter on the second command also override the values specified or defaulted for the first command.

Example 4: Changing the IPL Source After Immediate End

```
PWRDWNSYS  OPTION(*IMMED)  RESTART(*YES)  IPLSRC(A)
```

This command causes the system to end immediately and change the IPL source to A. When the system restarts, it IPLs on the A source.

QRYDOCLIB (Query Document Library) Command

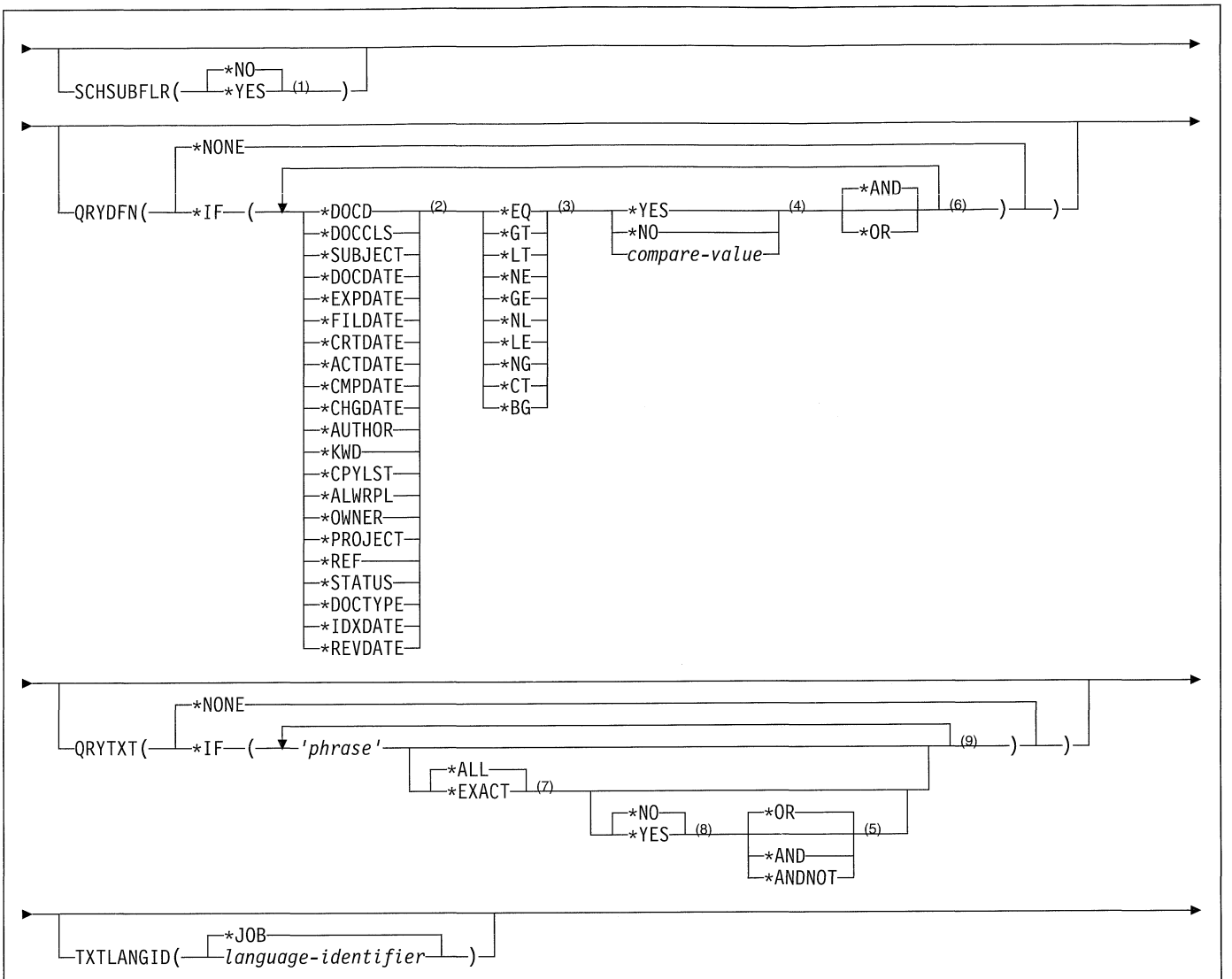


Notes:

P All parameters preceding this point can be specified in positional form.

1 A maximum of 20 repetitions

2 A maximum of 100 repetitions



Notes:

1 If SCHSUBFLR(*YES) is specified, FLR(*ALL) and FLR(*NONE) are not valid and only one user-specified folder name can be specified.

2 Profile parameter.

3 Relational operator.

4 Compare value.

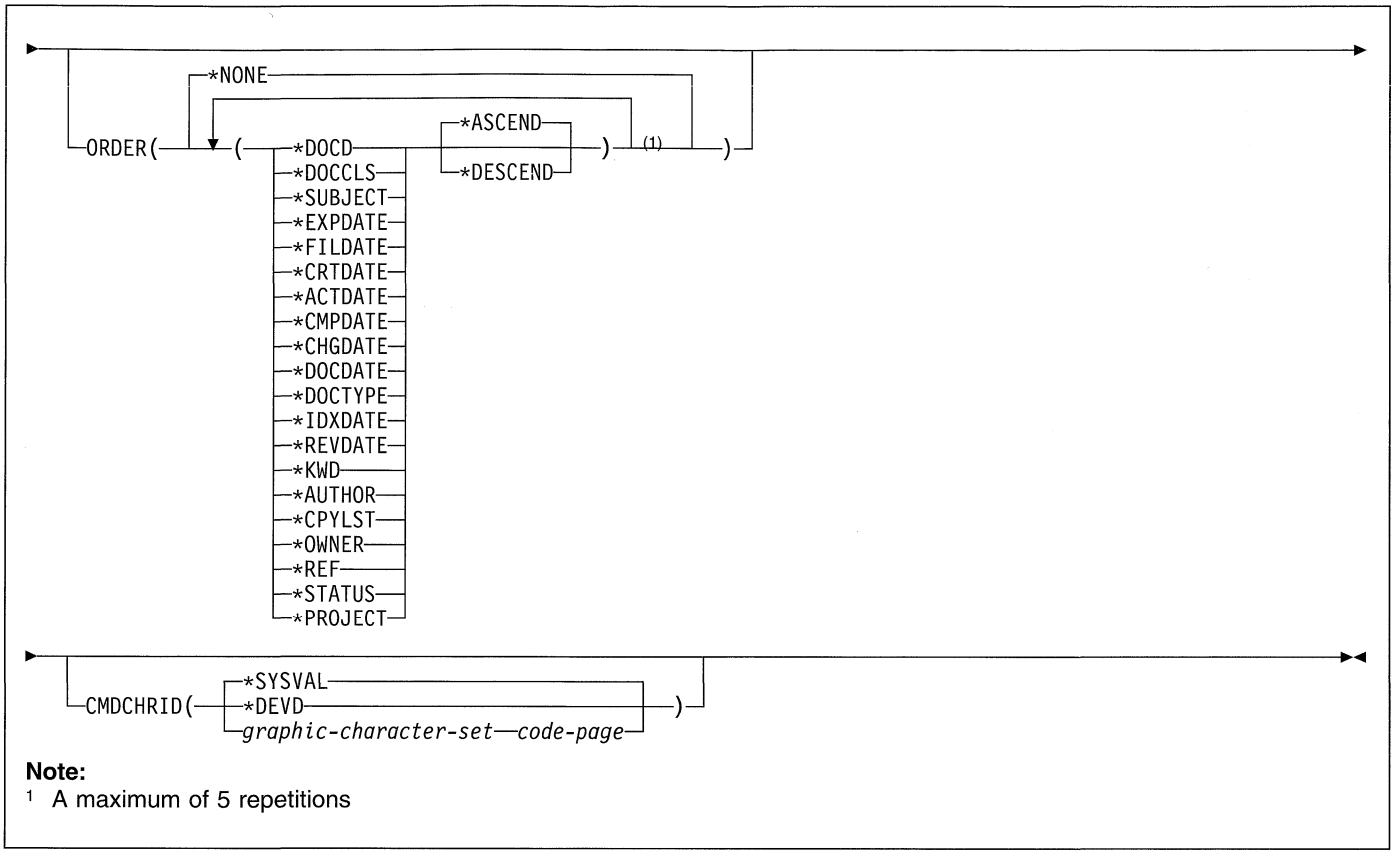
5 Logical operator to group multiple conditions.

6 A maximum of 49 repetitions

7 Type of phrase value.

8 Allow synonyms value.

9 A maximum of 30 repetitions



Note:
[1] A maximum of 5 repetitions

Purpose

The Query Document Library (QRYDOCLIB) command allows the user to search for documents within the document library to which the user is authorized and to copy information about the documents that satisfy the search request into a database file for processing.

When the QRYDOCLIB command is run, a document list object is created. The document list object is created regardless of whether an output file is produced. This document list object is used by the OfficeVision/400 product.

Restriction: The current user of this command must have the authority to work on behalf of the specified user ID address. To work on behalf of other users, the user must have special permission granted with the Grant User Permission (GRTUSRPMN) command. Several QRYDOCLIB commands can run concurrently. If the document list name or the output file is the same on more than one QRYDOCLIB command, errors may occur.

Note: The format of the output file must be the same as OSIQDL of the system file, QSYS/QAOSIQDL.

Optional Parameters

USRID

Specifies the user ID and address of the user for whom this request is made.

***CURRENT:** The user profile under which the current job is running is used.

Element 1: User ID

user-ID: Specify the user ID of the user for whom the documents are requested.

Element 2: User Address

user-address: Specify the user address of the user for whom the documents are requested.

OUTFILE

Specifies the name of the database file to which the output is directed. If the output file does not exist, this command creates a database file in the specified library. If the file is created by this function, the descriptive text is *OUTFILE created by QRYDOCLIB* and the authority for users without specific authority to the file is *EXCLUDE.

***NONE:** The output is not directed to a database file. A message is returned to the user indicating the number of documents that satisfied the search request. More information on defining the format of database files (output files) is in the *Office Services Concepts and Programmer's Guide*.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the qualified name of the database file that receives the output. This file can be reused when other QRYDOCLIB commands are entered. Output from the file can start at the start of the file member or be added to the file, as specified in the OUTMBR parameter. The IBM-supplied database file, QSYS/QAOSIQDL, cannot be specified.

OUTMBR

Specifies the name of the database file member to which the output is directed. If a member already exists, the system uses the second element of this parameter to determine whether the member is cleared before the new records are added. If the member does not exist and a member name is not specified, the system creates a member with the name of the output file specified on the OUTFILE parameter. If an output file member name is specified, but the member does not exist, the system creates it.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter. If the member exists, the system adds records to the end of the member or clears the member and then adds the records.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

OUTDTATYP

Specifies the type of information about the selected documents that is written to the output file if one is specified on the OUTFILE parameter.

***DFT:** A system-created name is used as the default name. The document information record is written to the output file. Specifying OUTDTATYP(*DFT) is equivalent to specifying OUTDTATYP(*DOCD). The following record is written to the output file:

Record Code	Description
105	Document Description

***ALL:** All record formats about the document are written to the output file. These record formats include the following:

Record Code	Description
105	Document Description
110	Creation Date
115	Expiration date
120	Document date
125	File date
130	Change date
135	Action due date
140	Completion date
145	Author
150	Copy list
155	Document class
160	File cabinet reference
165	Subject
170	Keyword
175	Reference
180	Status
185	Project
190	Last indexed date
195	Last revision date
500	Interchange document profile data

***DOCD:** The document description record is written to the output file.

***DOCCLS:** The document class record is written to the output file.

***SUBJECT:** The subject records are written to the output file.

***EXPDATE:** The expiration date record is written to the output file.

***FILDATE:** The file date record is written to the output file.

***CHGDATE:** The date of the last changed record is written to the output file.

***CRTDATE:** The create date record is written to the output file.

***ACTDATE:** The action due date record is written to the output file.

***CMPDATE:** The completion date record is written to the output file.

***DOCDATE:** The document date record is written to the output file.

***AUTHOR:** The author records are written to the output file.

***KWD:** The keyword records are written to the output file.

***CPYLST:** The copy list records are written to the output file.

***REF:** The reference record is written to the output file.

***STATUS:** The status record is written to the output file.

***PROJECT:** The project record is written to the output file.

***FILCAB:** The file cabinet reference record is written to the output file.

***IDXDATE:** The last indexed date record is written to the output file. OfficeVision/400 text search services must be installed if this value is specified.

***REVDATE:** The date of the last revision to the document content is written to the output file.

***IDP:** The interchange document profile is written to the output file.

DOCL

Specifies the name of the document list. A document list is an object that contains a pointer to each document in the document library that the current user is authorized to request. This list is a copy of the library at the time the search was run. As documents are deleted from or added to the library, the document list is not updated. The document library list name is specified with the name of the user requesting the search. Users can use identical document names. For example, Tom could name his list SALES and so could Mary. The system knows the lists as TOM_SALES and MARY_SALES.

***DFT:** A system-created name is used as the default document list name. The default list is the same as the user ID entered in the USRID parameter (TOM_TOM or MARY_MARY).

***NONE:** No document list is created.

document-list-name: Specify the name of the document list. Up to 8 characters can be used.

TEXT

Specifies text that briefly describes the document list object. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

TIMLMT

Specifies the maximum run time (in minutes) the search request can use.

***NOMAX:** No time limit for the search is set. All qualified documents are searched.

time-limit: Specify the maximum time limit (in minutes) that the search runs. Up to 4 digits, ranging from 1 through 9999, can be specified for the number of minutes. A two-hour limit is specified as TIMLMT(120). If the search has not been completed by the time the limit is reached, the search ends with an informational message followed by a completion message. The document list (*DOCL) and/or the output file will contain the documents found within the specified time limit.

SELLMT

Specifies the number of documents to select in the search.

***NOMAX:** No document limit for the search is set. All qualified documents are selected.

selection-limit: Specify the maximum number of documents to select. A value ranging from 1 through 32,767 can be specified. If there are more query requests than the set limit, the document list and the output file contain the information about the selected documents up to this limit. If there is at least one more document that meets the query definition, an informational message is sent before the completion message.

FLR

Specifies the folders that are searched to locate the documents that match the query definition specified in the QRYDFN parameter.

***ALL:** All of the folders on the system are searched to locate the documents.

***NONE:** Documents not located in any folder are searched.

folder-name: Specify the name of the folders to search to locate the documents. A folder name can consist of a series of folder names (FLR1/FLR2/and so on) if the documents being searched for are located in a folder contained in another folder. A maximum of 100 folders can be specified and each folder name can be a maximum of 63 characters in length.

SCHSUBFLR

Specifies whether subfolders of the folder specified on the FLR parameter are searched.

***NO:** Subfolders are not searched.

***YES:** Subfolders of the specified folder are searched.

QRYDFN

Specifies whether a query definition is used to select documents. The values specified on this parameter are used to search the document library. If values other than *NONE are specified on both the QRYDFN parameter and QRYTXT parameter, only documents that match both sets of values are selected.

***NONE:** No query definition is used to select the documents. All documents that are owned or accessible are selected.

***IF:** A query definition is used to select the documents.

To specify the conditions under which documents are selected, a set of values is specified for each condition. Each set contains four values.

1. The name of the document profile parameter to be compared
2. One of the relational operator values
3. The comparison value
4. One of the logical operators, *AND or *OR

Values 1 and 3 are compared for the relationship specified by value 2.

Each QRYDFN relational set must be enclosed in parentheses. A maximum of 49 sets (conditions) can be specified.

Element 1: Profile Parameters

profile-parameter: Specify one of the profile parameters to be compared.

Value	Description
*DOCD	Document description
*DOCCLS	Document class
*SUBJECT	Document subject
*DOCDATE	Document date
*EXPDATE	Expiration date
*FILDATE	File date
*CRTDATE	Create date
*ACTDATE	Action due date
*CMPDATE	Completion date
*CHGDATE	Last date change
*DOCTYPE	Document type
*IDXDATE	Last indexed date
*REVDATE	Last revision date
*AUTHOR	Document author
*KWD	Keyword
*CPYLST	Copy list
*ALWRPL	Allow document replacement
*OWNER	Document owner
*PROJECT	Document project
*REF	Reference
*STATUS	Document status

Element 2: Relational Operator

relational-operator: Specify the operator that indicates the relationship that must exist between the profile parameter contents in the document and the value specified as the third part of this QRYDFN relation for the relation to be true. The operators that can be used are:

Value	Description
*EQ	Equal
*GT	Greater than
*LT	Less than
*NE	Not equal
*GE	Greater than or equal
*NL	Not less than
*LE	Less than or equal
*NG	Not greater than
*CT	Contains
*BG	Begins

The *CT operator performs a context search. It asks the system to determine whether the character string specified by this value is contained anywhere in the profile. For example, a query request of (*IF ((*SUBJECT *CT 'Sales'))) would find a match for subjects that were: '1985 Car Sales', 'Sales Awards', 'Salesperson Training Courses', 'Book of Sales Do's and Don'ts'.

The *BG operator performs a search that compares the specified value with the start of the profile parameter. The profile parameter is truncated or extended as necessary to match the length of the specified value. It asks the system to determine whether the character string specified by the value is contained at the start of the profile parameter. For example, a query request of (*IF ((*SUBJECT *BG 'Sales'))) would find a match for subjects of: 'Sales Awards', 'Salesperson Training Courses', and 'Sales by Phone'.

Some operators are excluded from being used with some profile parameters. If the excluded operators are specified in restricted profile parameters, the request is rejected with a diagnostic message followed by an escape message. Two cases are invalid:

- The *ALWRPL (allow document replacement) is a YES/NO value. The *EQ operator is the only operator allowed to have *ALWRPL.
- The *CT and *BG operators are not allowed with date fields such as CRTDATE and EXPDATE.

Element 3: Compare Values

compare-value: Specify the value to compare with the contents of the specified profile parameter. The parameter value is specified in apostrophes if it contains blanks or special characters.

The *ALWRPL field has two special values: *YES and *NO. When these are specified with the *ALWRPL field, they are changed to internal values for the indicator. When specified for the text field, *YES and *NO retain their original values.

The *OWNER field is an 8-character user ID followed by its address. Trailing blanks cannot be omitted from the user ID. For example, if the user ID is JMDOE and the address is SYSTEM1, the query request would be (*IF ((*OWNER *EQ 'JMDOE SYSTEM1'))). If the user ID is JIMSMITH, the query request would be (IF ((*OWNER *EQ 'JIMSMITHSYSTEM1'))).

Dates must be specified in the system date format.

The allowable length for the search fields is limited by the Document Interchange Architecture (DIA) search database. When the length of the value is greater than the maximum, the value is truncated to the allowed length. The maximum lengths are:

Value	Maximum Length
*DOCD	44 characters
*DOCCLS	16 characters
*SUBJECT	60 characters
*AUTHOR	20 characters
*KWD	60 characters
*CPYLST	60 characters
*OWNER	16 characters
*REF	60 characters
*STATUS	20 characters
*PROJECT	10 characters

For all operators except *CT and *BG, if a value that is shorter than the profile parameter value is specified, it is padded on the right with blanks to match the length of the profile parameter.

The case (upper, lower, or mixed) of the letter characters used to enter the original parameter value or the case of the comparison value does not matter. The

system changes both the specified comparison value and the original parameter value to upper case before making a comparison. If the original document had been filed with a subject equal to 'Salesperson Training Courses', any of the following would be a match:

```
(*IF ((SUBJECT *EQ 'salesperson training
  courses')))
(*IF ((SUBJECT *EQ 'Salesperson Training
  Courses')))
(*IF ((SUBJECT *EQ 'SALESPERSON TRAINING
  COURSES')))
(*IF ((SUBJECT *CT 'PERSON')))
(*IF ((SUBJECT *CT 'person')))
(*IF ((SUBJECT *BG 'SALES')))
(*IF ((SUBJECT *BG 'sales')))
```

Element 4: Logical Operator

***AND:** The profile parameter value relational groups on both sides of the *AND value must be satisfied before a document is selected.

***OR:** If the parameter value relational group on either side of the *OR value is satisfied, the document is selected.

The logical operators are used to group conditions. The first AND operator encountered signifies that a condition group starts with the condition immediately preceding the AND operator. Subsequent conditions with the AND operator are added to the condition group. The first condition encountered that contains the OR operator (when no more matrix entries exist) ends the condition group.

For example:

```
QRYDFN(*IF
  ((COND1 *OR)
  (COND2 *AND)  <----|
  (COND3 *AND)       |--Group 1
  (COND4 *AND)  <----|
  (COND5 *OR)
  (COND6 *OR)
  (COND7 *AND)  <----|
  (COND8 *AND)       |--Group 2
  (COND9)))     <----|
```

The previous example could be expressed:

```
(cond1) | (cond2 & cond3 & cond4 & cond5)
  | cond6 | (cond7 & cond8 & cond9)
```

Because the AND operator is used to group conditions, the following logical expression cannot be used by the QRYDFN parameter.

```
(cond1 | cond2) & (cond3 | cond4)
```

QRYDFN Examples

```
QRYDFN(*IF ((*ACTDATE *GE '6/1/89' *AND)
  (*AUTHOR *EQ 'JOHN HARKER' *OR)
  (*ACTDATE *GE '6/1/89' *AND)
  (*PROJECT *EQ 'MGMT')))
```

All documents that have an action date later than or equal to 6/1/89 with author John Harker or project MGMT are selected. This request is made up of two condition groups (AND sets). The first group requires that the documents selected 1) must have an action date of 6/1/89 or later and 2) must have John Harker as the author. The second group requires that the documents selected 1) must have an action date of 6/1/89 or later and 2) must be part of project MGMT. If either of these condition groups is true, the document is selected.

```
QRYDFN (*IF ((*AUTHOR *EQ 'SUSAN MICKLE' *OR)
  (*PROJECT *EQ 'BASEBALL' *AND)
  (*CMPDATE *GE '6/1/89')))
```

All documents that with the author of Susan Mickle or that have a project of BASEBALL with a completion date on or after 6/1/89 are selected.

```
FLR('RECORDS/SPORTS/BASEBALL/NATIONAL')
  QRYDFN(*IF ((*DOCD *CT 'giants' *AND)
             (*FILDATE *GE '1/1/84' *AND)
             (*FILDATE *LE '10/1/86')))
```

If the word 'giants' is contained somewhere in the document description profile parameter, and if the document file date is between 1/1/84 and 10/1/86, the document is selected.

The NATIONAL folder is contained in the BASEBALL folder, which is in the SPORTS folder, which is contained in the RECORDS folder.

```
QRYDFN(*IF ((*EXPDATE *LE '1/1/86')))
```

All documents accessible by the user doing the search where the expiration date is less than or equal to 1/1/86, are selected.

```
QRYDFN(*IF ((*AUTHOR *EQ 'ANDERSON' *AND)
  (*DOCCLS *EQ 'account 431-2' *AND)
  (EXPDATE *LE '1/1/86')))
```

All documents accessible by the user doing the search, when all three conditions on the author, document class, and expiration date profile parameters are met, are selected.

```
QRYDFN(*IF ((*KWD *EQ 'alphabet soup' *OR)
  (*KWD *CT Campbells *OR)
  (*KWD *BG 'good for you')))
```

All documents accessible by the user doing the search, when any one of the three keyword tests is satisfied, are selected.

QRYTXT

Specifies the text search values used to select documents. The values specified on this parameter are used to search the text search index. If values other than *NONE are specified on both the QRYDFN parameter and the QRYTXT parameter, only documents that match both sets of values are selected.

***NONE:** No text search values are used to select the documents.

***IF:** Text search values are used in the document search.

Note: When *IF is specified, ordering is not allowed. *NONE must be specified on the ORDER parameter.

To specify the conditions under which documents are selected, a set of values is specified for each condition. Each set contains four values:

1. A phrase, which the system compares to entries in the text search index
2. One of the type of phrase matching values
3. One of the allow synonyms values
4. One of the logical operators

A maximum of 30 sets of values can be specified. Each set must be enclosed in parentheses.

Element 1: Phrase

'phrase': Specify a phrase of one or more words, which the system compares to entries in the text search index. A maximum of 50 characters can be specified. When specifying phrases:

- an asterisk (*) can be used to mask a whole word within a phrase. For example, when searching for documents containing references to various annual reports, the following phrase can be specified:

  ```
  annual * report
  ```

  The search results will include documents containing such phrases as annual budget report, annual progress report, and annual sales report. The search results will also include documents containing the phrase 'annual report' without a word in between.

  When using a word mask, a word must be specified before and after the asterisk. A word mask at the beginning or end of a phrase is ignored.

- an asterisk (*) can be used to mask part of a word within a phrase. The mask can be used at the beginning, middle, or end of a word. For example, when searching for documents containing references to word processing, the following phrase can be specified:

  ```
  word process*
  ```

  The search results will include documents containing such phrases as word processing, word processor, and word processed.

- a question mark (?) can be used to mask one or more characters in a word. For example, when searching for documents containing references to the various spellings of Johnson, the following phrase can be specified:

  ```
  j?hns?n
  ```

  The search results will include documents containing such phrases as Johnson, Johnsen, and Jahnson.

Element 2: Type of Phrase

***ALL:** The phrase must be contained within one sentence, but the words do not have to be in the specified order.

***EXACT:** The phrase must be contained within one sentence and the words must be in the specified order.

Element 3: Synonyms

***NO:** No synonyms are used when searching for documents.

***YES:** Synonyms for each word in the phrase, if available, are used to compare entries in the text index.

Note: Using synonyms may affect the performance of the request by causing more words to be searched for and by possibly causing more documents to be selected.

Element 4: Logical Operator

***OR:** If the phrase on either side of the *OR value is found, the document is selected.

***AND:** If the phrases on both sides of the *AND value are found, the document is selected.

***ANDNOT:** If the phrase following the *ANDNOT value is not found, the document is selected.

TXTLANGID

Specifies the language identifier of the document for which the user is searching. This parameter is not allowed if the QRYTXT parameter is not specified.

***JOB:** The language identifier specified for the job in which this command is entered is used.

language-identifier: Specify the language identifier.

ORDER

Specifies the order in which selected documents are placed in the created document list object and the output file (if OUTFILE is specified). For example, if *SUBJECT is specified, the selected documents are returned in order, based on the collating sequence of the document subject.

When a user specifies an order to the search request, the performance of the request may be affected. The request performs best if an order is not specified. Up to 5 document profile parameters can be specified.

Element 1: Method of Order

***NONE:** No order is applied to the selected documents.

***DOCD:** The returned documents are ordered by the document name profile parameter.

***DOCCLS:** The returned documents are ordered by the document class profile parameter.

***SUBJECT:** The returned documents are ordered by the subject profile parameter.

***EXPDATE:** The returned documents are ordered by the expiration date profile parameter.

***FILDATE:** The returned documents are ordered by the file date profile parameter.

***CRTDATE:** The returned documents are ordered by the create date profile parameter.

***ACTDATE:** The returned documents are ordered by the action due date profile parameter.

***CMPDATE:** The returned documents are ordered by the completion date profile parameter.

***CHGDATE:** The returned documents are ordered by the last changed date profile parameter.

***DOCDATE:** The returned documents are ordered by the document date profile parameter.

***DOCTYPE:** The returned documents are ordered by the document type profile parameter. Valid values range from 2 through 65535.

***IDXDATE:** The returned documents are ordered by the last indexed date profile parameter. Text search services must be installed if this value is specified.

***REVDATE:** The returned documents are ordered by the last content revision date.

***KWD:** The returned documents are ordered by the keyword profile parameter.

***AUTHOR:** The returned documents are ordered by the author profile parameter.

***CPYLST:** The returned documents are ordered by the copy list profile parameter.

***OWNER:** The returned documents are ordered by the owner profile parameter.

***REF:** The returned documents are ordered by the reference profile parameter.

***STATUS:** The returned documents are ordered by the status profile parameter.

***PROJECT:** The returned documents are ordered by the project profile parameter.

Element 2: Ascending or Descending Order

***ASCEND:** The returned documents are ordered in the ascending collating sequence.

***DESCEND:** The returned documents are ordered in the descending collating sequence.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the *Guide to Programming Displays*.

Note: The CMDCHRID parameter applies to the following parameters and means that the data is translated to the code page and character set that is common to all of the documents in the search database.

This value translates the DOCL, QRYDFN, USRID, QRYTEXT, and TEXT parameters to values for character set and code page of '697 500'.

This value translates the USRID parameter to character set and code page of '930 500'. The *Communications: Distribution Services Network Guide* contains the character set and code page table for '930 500'.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values used to create the command parameter.

Element 2: Code Page

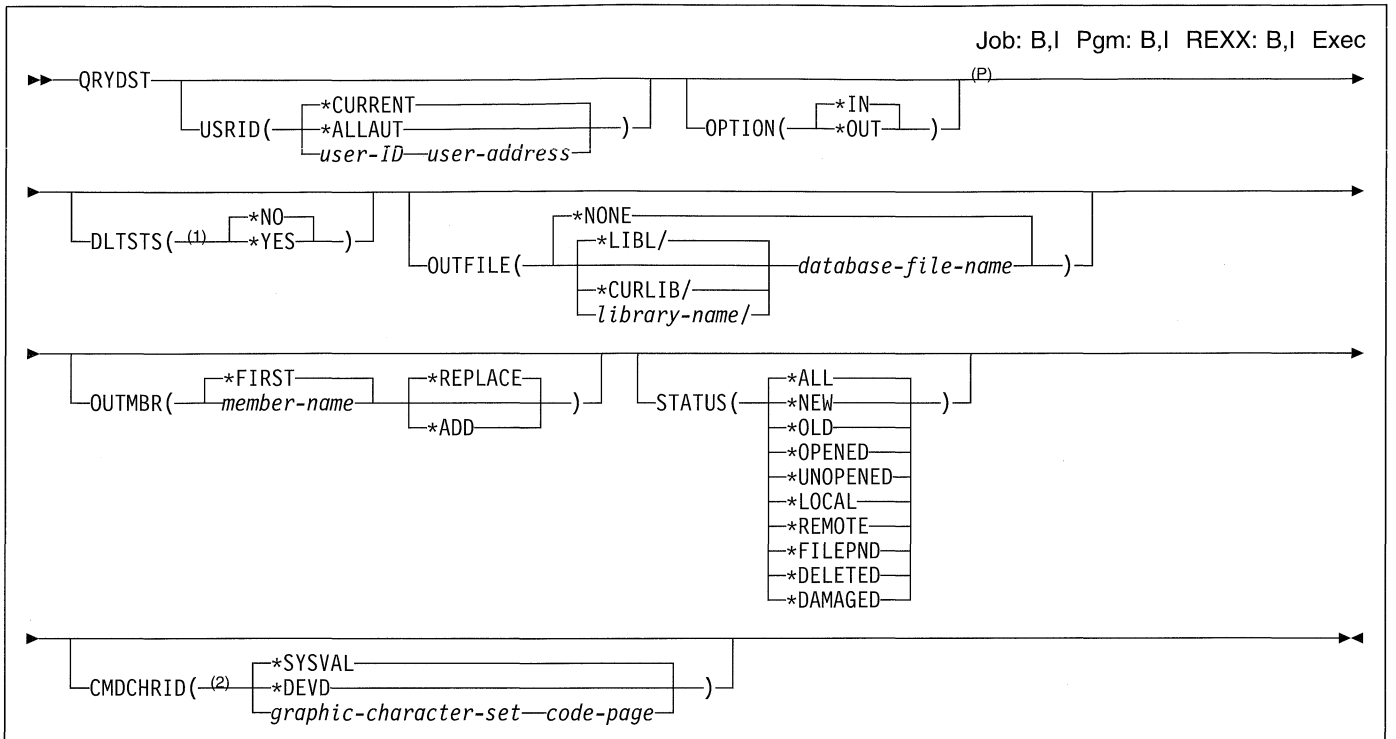
code-page: Specify the code page value used to create the command parameters. Valid values range from 1 through 999.

Example

```
QRYDOCLIB  USRID(*CURRENT) OUTFILE(*NONE)
  QRYDFN(*IF ((*DOCD *EQ DOCDESC *AND)
  (*DOCCLS *BG CLASS *OR) (*FILDATE *LE '06/13/88')))
  DOCL(MYLIST)
```

This command searches for documents that meet the following search conditions: document description equals DOCDESC and document class starts with Class; or the file date is on or before 06/13/88. The results of the search will be stored in the document list MYLIST.

QRYDST (Query Distribution) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 DLTSTS does not apply to incoming distributions. When OPTION(*IN) is specified, the DLTSTS parameter is ignored.

2 This parameter value applies to the USRID parameter.

Purpose

The Query Distribution (QRYDST) command allows a request for distribution information for the user or on behalf of another user.

Restrictions:

1. If the current user of this command requests distribution information for another user, the current user must have been granted the authority to work on behalf of the other user by means of the Grant User Permission (GRTUSRPMN) command.
2. If USRID(*ALLAUT) is specified and the current user of this command does not have the authority to work on behalf of the other user, only the information about the current user's distributions is returned.
3. DLTSTS does not apply to incoming distributions. When OPTION(*IN) is specified, the DLTSTS parameter is ignored.
4. The requester of the command (the user who is signed on) must be enrolled in the system distribution directory.
5. Personal distribution cannot be questioned if the requester is working on behalf of another user.

Note: The formats of the output files must be OSLIN or OSLOUT. These formats are defined in the physical files QAOSILIN or QAOSILOT, respectively. These files are located in library QSYS. Users can specify the Create Duplicate Object (CRTDUPOBJ) command to create duplicates of these files for their libraries. If the user's library does not contain the files, the files are created when the command is run.

Optional Parameters

USRID

Specifies the user ID and address of the user making this request. The user specifying this command must have authority to work on behalf of the user specified in this parameter if the named user ID and address differs from that of the current user.

***CURRENT:** The user profile under which the current job is running is used.

***ALLAUT:** Distribution information is returned for users who have given the current user of this command the authority to work on their behalf.

Element 1: User ID

user-ID: Specify the user ID of the user for whom the distribution information is returned.

Element 2: User Address

user-address: Specify the user address of the user for whom the distribution information is returned.

OPTION

Specifies the type of distribution information that is returned.

***IN:** Information about incoming distributions is returned. If an output file is specified in the OUTFILE parameter, one incoming distribution information record per distribution is written to the output file.

***OUT:** Information about outgoing distributions is returned. If an output file is specified, N outgoing distribution information records per distribution are written to the output file. N is the number of original receivers of the distribution or the number of receivers that have distribution errors.

DLTSTS

Specifies whether the status being kept for outgoing distributions is deleted from the system. This can be error or confirmation of delivery status.

***NO:** The distribution status is not deleted from the system. The information is kept by the system and can be returned by a request using the QRYDST command.

***YES:** The distribution status is deleted if all receivers are at returned status or completed confirmation of delivery status.

Note: Other products use this status information. Care should be taken not to delete information used by other products to track their distributions.

Status is deleted by the system if all receivers have returned status or the status is returned to another product (such as the OfficeVision/400) for this user.

OUTFILE

Specifies the qualified name of the database file to which the output of the command is directed. If the file does not exist, this command creates a database file in the specified library.

For incoming distributions, the system uses QAOSILIN in QSYS with the format name of OSLIN as a model.

For outgoing distributions, the system uses QADSILOT in QSYS with the format name of OSLOUT as a model.

The authority for users with no specific authority to the output file is *EXCLUDE. More information on defining the format of database files (output file) is in the *Office Services Concepts and Programmer's Guide*.

***NONE:** The output is not directed to a database file. If *NONE is specified, the output from this command is a completion message containing the number of distributions on the DIA Distribution Recipient Index (*DRX) for the specified category and user.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the qualified name of the database file that receives the output of the display.

OUTMBR

Specifies the name of the database file member to which the output is directed. If a member already exists, the system uses the second element of this parameter to determine whether the member is cleared before the new records are added. If the member does not exist and a member name is not specified, the system creates a member with the name of the output file specified on the OUTFILE parameter. If an output file member name is specified, but the member does not exist, the system creates it.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

STATUS

Specifies the mail entry status of the distribution for which the user is requesting information. This parameter is valid only if *IN is specified on the OPTION parameter and an output file is specified on the OUTFILE parameter.

***ALL:** Distribution information is returned regardless of the distribution's mail entry status.

***NEW:** Distribution information is returned only for distributions with a mail entry status of NEW.

***OLD:** Distribution information is returned only for distributions with a mail entry status of OLD. A mail entry status of OLD indicates that the distribution has been queried once but has not been processed.

***OPENED:** Distribution information is returned only for distributions with a mail entry status of OPENED.

***UNOPENED:** Distribution information is returned for distributions with a mail entry status of OLD or NEW.

***LOCAL:** Distribution information is returned only for distributions with a mail entry status of LOCAL. A mail

entry status of LOCAL indicates that the distribution has been filed on the local system.

***REMOTE:** Distribution information is returned only for distributions with a mail entry status of REMOTE. A mail entry status of REMOTE indicates that the distribution has been filed on a remote system.

***FILEPND:** Distribution information is returned only for distributions with a mail entry status of FILEPND. A mail entry status of FILEPND indicates that the distribution is being filed on a local or remote system but the filing has not been completed.

***DELETED:** Distribution information is returned only for distributions with a mail entry status of DELETED. A mail entry status of DELETED indicates that the document referred to by the distribution has been deleted.

***DAMAGED:** Distribution information is returned only for distributions with a mail entry status of DAMAGED. A mail entry status of DAMAGED indicates that the document referred to by the distribution is damaged.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the *Guide to Programming Displays*.

Note: This value translates the USRID parameter to the character set and code page set of '930 500'. The *Communications: Distribution Services Network Guide* contains the character set and code page table for '930 500'.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values used to create the command parameter.

Element 2: Code Page

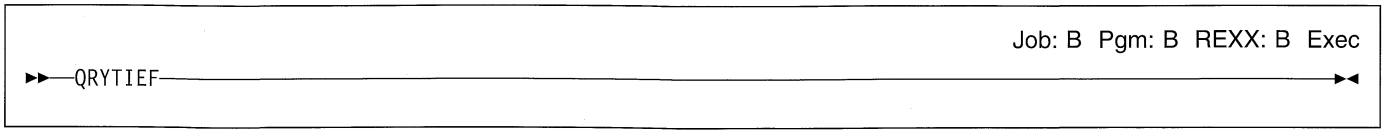
code-page: Specify the code page. Valid values range from 1 through 9999.

Example

```
QRYDST  USER(*CURRENT)  OPTION(*IN)
  OUTFILE(*CURLIB/MYFILE)  OUTMBR(*FIRST *ADD)
```

This command requests information about incoming distributions for the current user of this command. The output is directed to the database file MYFILE in the user's current library and is added to the first member in that file.

QRYTIEF (Query Technical Information Exchange File) Command



Purpose

The Query Technical Information Exchange File (QRYTIEF) command allows the user to determine whether any files are available to be received from the remote support network. A message is returned that specifies the size (number of records) of the largest file that is to be received.

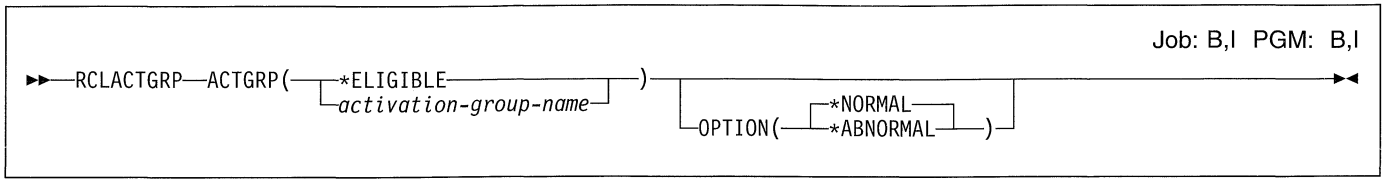
There are no parameters for this command.

Example

```
QRYTIEF
```

This command sends a message that specifies the number of files to be received from the remote support network and the size of the largest file to be received.

RCLACTGRP (Reclaim Activation Group) Command



Purpose

The Reclaim Activation Group (RCLACTGRP) command frees the resources associated with a specified activation group. This command can reclaim a specific activation group or all eligible activation groups.

An activation group is eligible to be reclaimed if it meets the following criteria:

- The activation group is not the default activation group.

  The default activation group cannot be reclaimed.

- The activation group is not active.

  An activation group cannot be reclaimed if there are programs or procedures running within the activation group.

- The activation group is not one of the debug activation groups.

  When the job is in debug mode, the activation groups in use do not appear as active on the Call Stack or Display Activation Group displays. When the job is no longer in debug mode, the activation groups used can be reclaimed.

When an activation group is reclaimed, all resources within the scope of the activation group are reclaimed. Resources within the scope of the activation group include static storage for programs in the activation group, open files, user interface manager (UIM) application resources, Common Programming Interface (CPI) Communications conversations, hierarchical file systems (HFS) resources, open FM sessions, and pending changes for the commitment definition.

A close option can be specified on this command, and is used when closing mixed, communications, binary synchronous (BSC), and ICF files. If an activation group level commitment definition has been started for the activation group, and it has pending committable changes, the close option also indicates whether the system implicitly commits or rolls back the pending changes before ending the commitment definition. When specifying a close option of *NORMAL, and there are no errors when closing files using the activation group level commitment definition, a commit is performed. Otherwise, a rollback is performed.

Required Parameter

ACTGRP

Specifies the activation group to be reclaimed.

***ELIGIBLE:** All eligible activation groups within the job are deleted.

activation-group-name: Specify the activation group to be reclaimed. The activation group can only be reclaimed if it has no active calls. If active calls exist, a message is displayed informing the user that the request failed. If the activation group is not found, a message is displayed informing the user that the request failed because the activation group was not found.

Optional Parameter

OPTION

Specifies whether to commit or roll back pending changes for an activation group level commitment definition, and whether a normal or abnormal close notification is sent to the attached host system when mixed, communications, BSC, and ICF files are closed. This parameter is ignored for all other files and objects within the scope of the activation group.

***NORMAL:** The changes pending for an activation group level commitment definition are committed (if there are no errors when closing files using the commitment definition), and a normal close notification is sent to the attached host system when mixed, communications, BSC, and ICF files are closed.

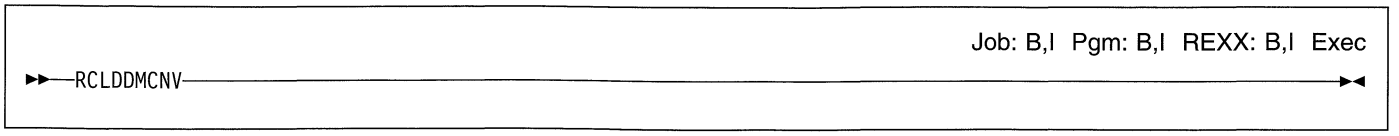
***ABNORMAL:** The changes pending for an activation group level commitment definition are rolled back and an abnormal close notification is sent to the attached host system when mixed, communications, BSC, and ICF files are closed.

Example

```
RCLACTGRP  ACTGRP(MYGROUP)
```

This command reclaims the activation group MYGROUP.

RCLDDMCNV (Reclaim Distributed Data Management Conversations) Command



Purpose

The Reclaim Distributed Data Management Conversations (RCLDDMCNV) command reclaims all distributed data management (DDM) source system conversations that are not currently being used by a source job, even if the DDMCNV attribute value for the job is *KEEP. The command allows the user to reclaim unused DDM conversations without closing all open files or doing any of the other functions performed by the RCLRSC command. The RCLDDMCNV command applies only to the DDM conversations for the job on the *source* system in which the command is entered.

Although this command applies to *all* DDM conversations used by a job, using it does *not* mean that all of them are taken down. A conversation is taken down *only* if there are no active users for that conversation.

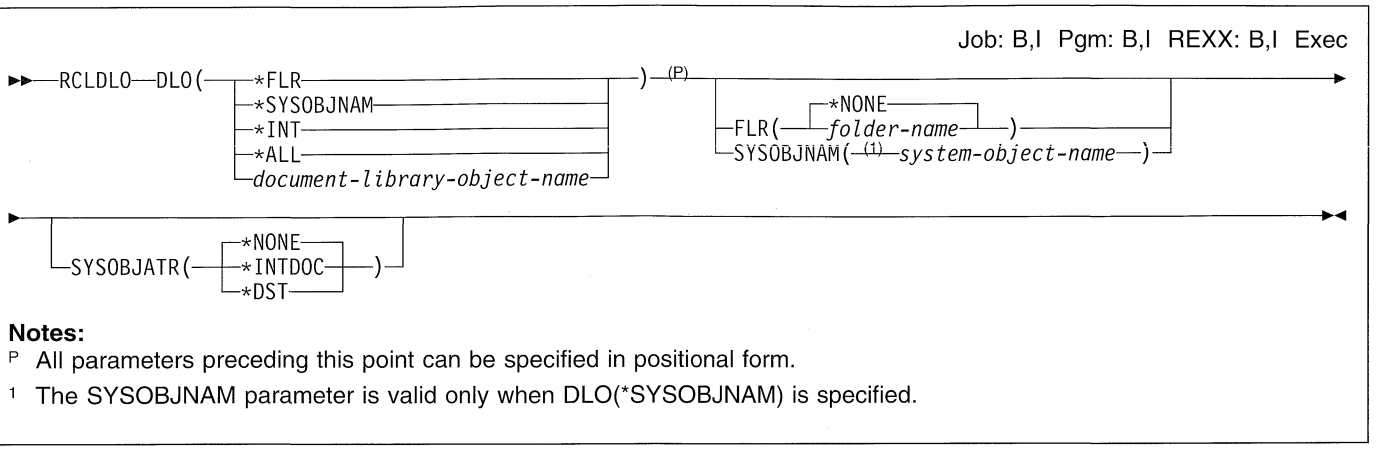
There are no parameters for this command.

Example

```
RCLDDMCNV
```

This command checks all DDM conversations for the job in which the command is entered, determines if there are any users of each conversation, and reclaims each one not being used.

RCLDLO (Reclaim Document Library Object) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 The SYSOBJNAM parameter is valid only when DLO(*SYSOBJNAM) is specified.

Purpose

The Reclaim Document Library Object (RCLDLO) command is used to reclaim:

- A document
- A folder
- A folder and all documents and folders directly or indirectly contained within it
- Internal document library system objects
- Internal document library system objects and all folders and documents on the system, including synchronizing their document details

Restrictions: (1) The user does not need authority to a document or folder to reclaim it. (2) Exclusive use of the document or folder is required while it is being reclaimed. (3) The user does not need to be enrolled in the system distribution directory. (4) Reclaiming internal document library system objects or all document library objects can only be done when no other jobs are using folders or documents. The user of this command may receive an error message indicating that internal objects are locked. This means that another user is using document library functions which cannot run at the same time as the RCLDLO command; retry this command after other document library activity has ended.

Note: AS/400 PC/Support jobs and several jobs which run in the QSNADS subsystem work with documents and folders. These jobs can prevent RCLDLO DLO(*ALL) or RCLDLO DLO(*INT) from being run, due to locks they obtain on document library objects. All jobs running in the QSNADS subsystem and AS/400 PC/Support jobs should be ended before attempting to run RCLDLO DLO(*ALL) or RCLDLO DLO(*INT).

Required Parameters

DLO

Specifies the document library object to reclaim.

***FLR:** A folder, and all folders and documents directly or indirectly within it, are to be reclaimed.

***SYSOBJNAM:** A system object name is used to identify the folder or document to reclaim. DLO(*SYSOBJNAM) must be specified when reclaiming a folderless document, an internal document, or a distribution document. It can also be specified instead of a folder or document name if the system object name is known.

***INT:** Internal document library system objects are to be reclaimed.

Note: The internal document library system objects are used to manage the documents and folders on the AS/400. RCLDLO DLO(*INT) is only necessary if the internal objects become damaged. If the internal objects are damaged, attempts to access documents and folders will result in the message CPF8A46 (Internal system objects are damaged), possibly followed by the message CPF9032 (Document interchange session not started),

***ALL:** Internal document library system objects and all documents and folders are to be reclaimed. DLO(*ALL) synchronizes the relationships between all document library objects and their document details and can be used to fix inconsistencies between them.

Note: The RCLDLO DLO(*ALL) command can be a long-running function, depending on the number of documents and folders on the system. If the RCLDLO command can be issued at the user's discretion, the user may wish to avoid the operation until the time required can be scheduled.

document-library-object-name: Specify the name of the folder or document to reclaim.

Optional Parameters

FLR

Specifies the name of the folder that contains the document.

Note: A folder name is entered in this parameter only if *FLR, or a folder or document name is entered in the DLO parameter.

***NONE:** The folder or document is not within a folder.

folder-name: Specify the name of the folder that contains the document or folder to reclaim, or the name of the folder to reclaim along with all documents and folders within it.

SYSOBJNAM

Specifies the system object name. This parameter is valid only when DLO(*SYSOBJNAM) or DOCL(*SYSOBJNAM) is specified. A full ten characters must be specified.

SYSOBJATR

Specifies the attributes of the object to reclaim. A value other than *NONE can be entered in this parameter only if *SYSOBJNAM is entered in the DLO parameter.

***NONE:** The object to reclaim is a filed document or folder.

***INTDOC:** The object to reclaim is an internal document.

***DST:** The object to reclaim is a distribution document.

Examples

Example 1: Reclaiming a Folder

```
RCLDLO  DLO(FLR1)
```

This command reclaims folder FLR1.

Example 2: Reclaiming a Document Within a Folder

```
RCLDLO  DLO(A)  FLR(FLR2)
```

This command reclaims folder or document A in folder FLR2.

Example 3: Reclaiming a Folder and All Documents and Folders Within It

```
RCLDLO  DLO(*FLR)  FLR(FLR3)
```

This command reclaims folder FLR3 and all folders and documents directly or indirectly contained within it.

Example 4: Reclaiming an Internal Document

```
RCLDLO  DLO(*SYSOBJNAM)  SYSOBJNAM(AMBT133080)
  SYSOBJATR(*INTDOC)
```

This command reclaims the internal document specified by the system object name AMBT133080.

Example 5: Reclaiming a Distribution Document

```
RCLDLO  DLO(*SYSOBJNAM)  SYSOBJNAM(AMBT133082)
  SYSOBJATR(*DST)
```

This command reclaims the distribution document specified by the system object name AMBT133082.

Example 6: Reclaiming Document Library System Objects

```
RCLDLO  DLO(*INT)
```

This command reclaims internal document library system objects.

Example 7: Reclaiming Document Library System Objects and All Documents and Folders

```
RCLDLO  DLO(*ALL)
```

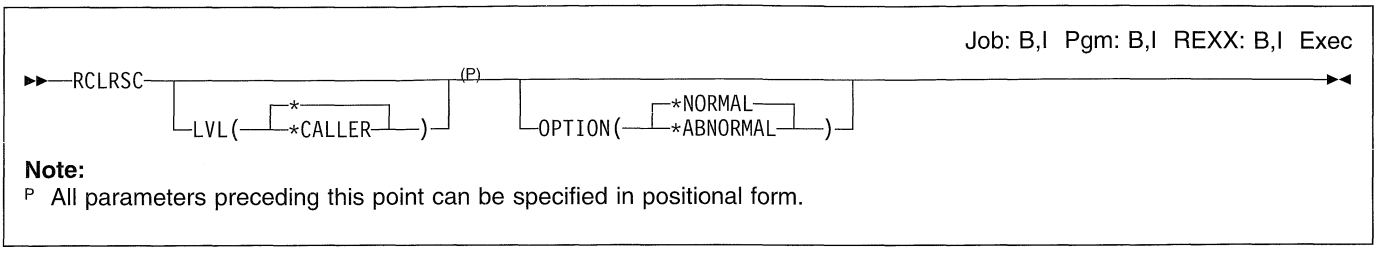
This command reclaims internal document library system objects and all documents and folders and synchronizes their document details.

Additional Considerations

When RCLDLO DLO(*ALL) is run, the following items indicate what happens for folders and documents whose document details are updated.

- Objects that have lost folder path information are associated with a system reclaim folder. An informational message is sent to QSYSOPR message queue for each object reclaimed in this manner. System reclaim folders, recognized by the name 'QRCLnnnn.DOC' or 'QRCLnnnn.FLR,' should be deleted from the system after moving the objects within them to an appropriate user folder. It may also be necessary to use the Edit Document Library Object Authority (EDTDLOAUT) command to grant specific authority to the reclaimed objects again.
- A log of the actions taken by the RCLDLO DLO(*ALL) command is sent to the joblog and a listing of status messages is sent to the QHST system log. Additional information about the objects shown in the log can be displayed by the Display Object Description (DSPOBJD) command.

RCLRSC (Reclaim Resources) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Reclaim Resources (RCLRSC) command is designed to free the resources for programs or procedures that are no longer active. The resources reclaimed by this command are static storage, open files, user interface manager (UIM) application resources, Common Programming Interface (CPI) Communications conversations, hierarchical file systems (HFS) resources, and open FM sessions. This command is normally used only in the controlling program of the application.

For procedures in ILE languages, the resources that can be reclaimed are those within the scope of the activation group in which the procedure is run.

- If the activation group is still in use, the resources cannot be reclaimed.
- If the activation group is no longer in use, the entire activation group is reclaimed. All resources within the scope of the activation group are reclaimed, and the activation group is ended.
- If an activation group level commitment definition has been started and it has pending committable changes, before the activation group is ended those changes will be committed or rolled back based on the close option specified.

For programs not written in ILE languages, the resources that can be reclaimed are those within the scope of the program activation. If the program is no longer active, the resources are reclaimed.

If the job is in debug mode, activation groups exist that do not appear to be active. These activation groups are not affected by the RCLRSC command. When the job is no longer in debug mode, the activation groups can be reclaimed.

When a program ends abnormally, this command can be used to reclaim resources with the close option *ABNORMAL. This close option is used when closing mixed, communications, binary synchronous (BSC), and intersystems communications function (ICF) files.

If an activation group level commitment definition has been started for an activation group to be ended, and it has pending committable changes, the close option also indicates whether the system implicitly commits or rolls back the pending changes before ending the commitment definition.

When specifying a close option of *NORMAL, and there are no errors when closing files using the activation group level commitment definition, a commit is performed. Otherwise, a rollback is performed.

The storage and other resources used by the opened files can then be used by other programs running on the system.

The RCLRSC command is *not* needed to reclaim the resources of all CL programs that end (return) normally, for RPG programs that have the last record (LR) indicator set on, or for COBOL programs. This is true because, on a normal return for all these types, the storage is freed and the files are closed automatically.

However, the RCLRSC command should *not* be used if it might be processed while any COBOL program is still active in the application; instead, the COBOL CANCEL statement should be used to reclaim the resources. That is, instead of using the RCLRSC command following a COBOL program that might end abnormally, the user should, in the COBOL program, use a CANCEL statement that can be processed when an abnormal condition occurs in that program.

The RCLRSC command does not close files that were opened with the Open Database File (OPNDBF) or the Open Query File (OPNQRYF) commands specifying TYPE(*PERM).

More information about how static storage is used is in the *CL Programmer's Guide* and the publications that apply to the high-level languages.

Restrictions:

1. Do not specify LVL(*CALLER) on this command if it is used in a CL program that also uses the Send File (SNDF), Receive File (RCVF), or Send Receive File (SNDRCVF) commands. Specifying RCLRSC LVL(*CALLER) in such a program causes unpredictable results when the SNDF, RCVF, or SNDRCVF commands are used after the program runs.
2. If an ILE procedure is run in the default activation group, the RCLRSC command cannot reclaim all the resources for the procedure. Open files can be closed, but static storage cannot be reclaimed.
3. Pending changes for activation group level commitment definitions are affected only for activation groups that are reclaimed. If the program or ILE procedure is running in the default activation group, pending changes to the

commitment definition are unaffected by this command. This job level commitment definition is also unaffected by this command.

Optional Parameters

LVL

Specifies the reference level at which resources are freed.

*: The reference level is the program or procedure that contains this RCLRSC command. The resources are reclaimed for programs or procedures that have finished running and returned control to this program.

***CALLER:** The reference level is the program or procedure that called the program or procedure containing this RCLRSC command. This value allows controlling programs or procedures written in a high-level language to call a CL program to reclaim resources to the level of the controlling program or procedure. The effect is the same as if the command were issued from the controlling program or procedure.

OPTION

Specifies whether a normal or abnormal close notification is sent to the attached host system when mixed, communications, BSC, and ICF files are closed. For activation groups reclaimed, the close option specifies whether to commit or roll back pending changes for activation group level commitment definitions. This parameter is ignored for all other files and objects within the scope of the activation group.

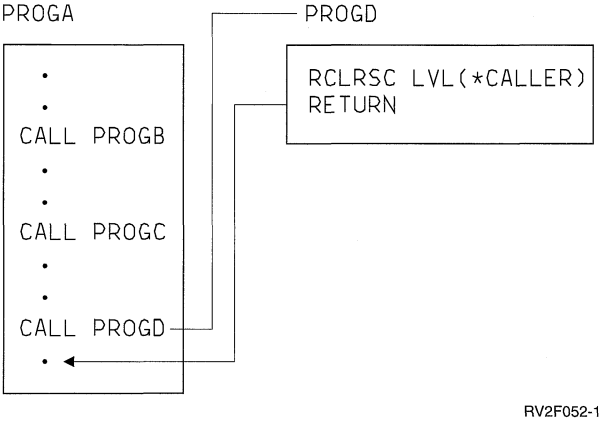
***NORMAL:** The attached host system is given a normal close notification when mixed, communications, BSC, and ICF files are closed. Pending changes to activation group level commitment definitions are committed if there are no errors when closing files using the commitment definition.

***ABNORMAL:** The attached host system is given an abnormal close notification when mixed, communications, BSC, and ICF files are closed. Use this when the controlling program detects error conditions that should be communicated to the host systems (the error condition need not be file-related). Pending changes to activation group level commitment definitions are rolled back.

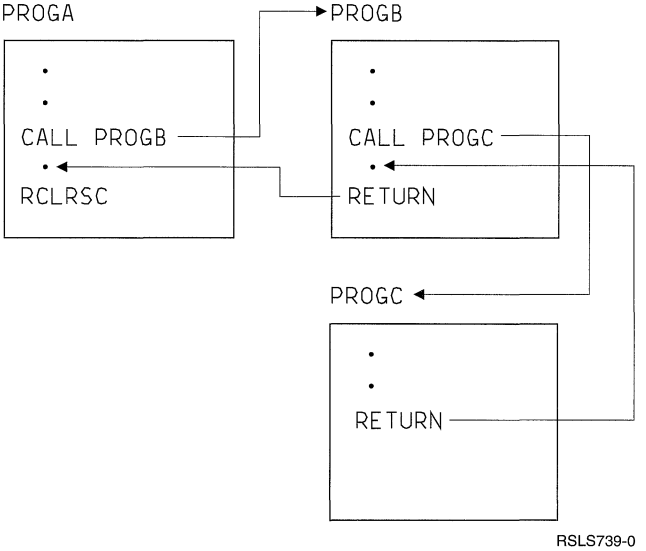
Examples

```
PROGA
      •
      •
CALL PROGB
RCLRSC
      •
      •
CALL PROGC
RCLRSC
      •
      •
      •
```

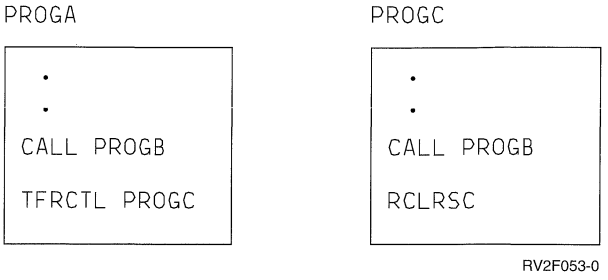
In this example, PROGA is a controlling program in an application. PROGA calls other programs, which return control to PROGA when they have finished running. Because control is returned to the next sequential instruction, the RCLRSC command is issued following each CALL command to free the static storage that was used by the called program, and to close the files that were left open.



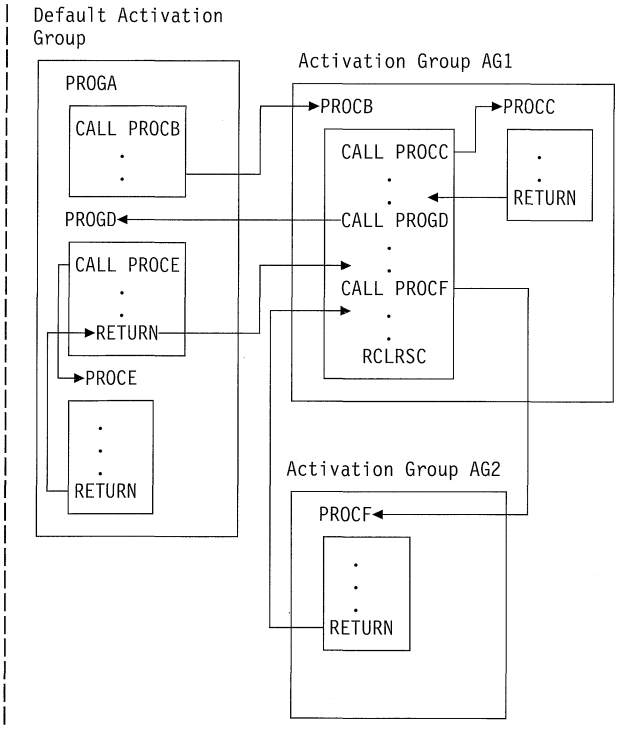
In this example, PROGA is a controlling program that is written in another high-level language. The RCLRSC command cannot be issued from the high-level language program so PROGD, a CL program, is called to issue the command. When the RCLRSC command is issued in PROGD, the static storage used by PROGB and PROGC is freed; files that were left open are closed.



In this example, PROGA is a controlling program. When the RCLRSC command is issued, the static storage used by PROGB and PROGC is freed; files that were left open are closed.



In this example, PROGA calls PROGB and, after returning from PROGB, PROGA transfers to program PROGC. Because PROGB has already been called, static storage exists, and the call to PROGB from PROGC does not cause any new allocation for static storage; PROGC cannot reclaim the static storage used by PROGB. If PROGB opened files when it was called by PROGA, these files would remain open; if PROGB opened files when it was called by PROGC, these files are closed.



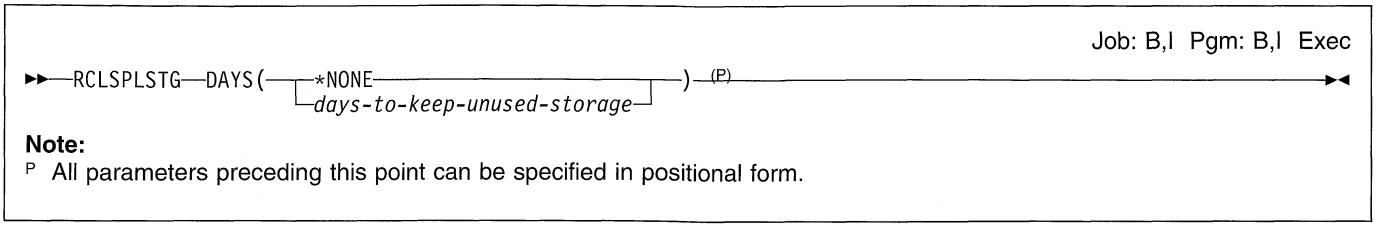
This example shows how ILE procedures and activation groups are affected by the RCLRSC command. You must be using one of the ILE languages in this example.

In this example, PROGA is a program running in the default activation group. PROGA calls ILE procedure PROCB, which causes activation group AG1 to be created. Procedure PROCB calls ILE procedure PROCC, which also runs in activation group AG1. After returning from PROCC, PROCB calls program PROGD which runs in the default activation group. Program PROGD calls ILE procedure PROCE, which also runs in the default activation group. After returning from PROCE, program PROGD returns to procedure PROCB. Procedure PROCB then calls ILE procedure PROCF, which causes activation group AG2 to be created. After returning from PROCF, procedure PROCB calls the RCLRSC command.

Procedure PROCB calls the RCLRSC command. Any resources in use by PROCB are still open. All resources used by PROCC are within the scope of the activation group AG1. Since AG1 is still in use, any resources used by PROCC are also still open. Any resources used by program PROGD are reclaimed. Since procedure PROCE ran in the default activation group and the RCLRSC command cannot reclaim all resources for ILE procedures which run in the default activation group, some of the resources used by PROCE are still open. Procedure PROCF ran in activation group AG2, and any resources used are within the scope of the activation group AG2. Since activation group AG2 is no longer in use, activation group AG2 is deleted and any resources within its scope are reclaimed.

Any other use of the RCLRSC command can result in files remaining open and storage remaining allocated.

RCLSPLSTG (Reclaim Spool Storage) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Reclaim Spool Storage (RCLSPLSTG) command reclaims unused storage for spooled files that have not been used for more than the number of days specified by the user. Spooled files are stored with database file members on the system. When a spooled file is deleted, the member is emptied but not deleted. Therefore, the member can be reused for the next spool file created. Reusing empty members improves the performance time when creating new spooled files. The RCLSPLSTG command deletes unused and empty database members. This command uses synchronous processing. More information about synchronous processing is in the *Advanced Backup and Recovery Guide*.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Required Parameters

DAYS
Specifies that storage for spooled files that have not been used for the number of days specified by the user is reclaimed.

***NONE:** All unused storage for spooled files is reclaimed. If there is no unused storage when subsequent spooled files are created, then the system allocates storage for those spooled files. The allocation of storage for these spooled files may slow down the jobs that are creating them. More information is in the *Data Management Guide*.

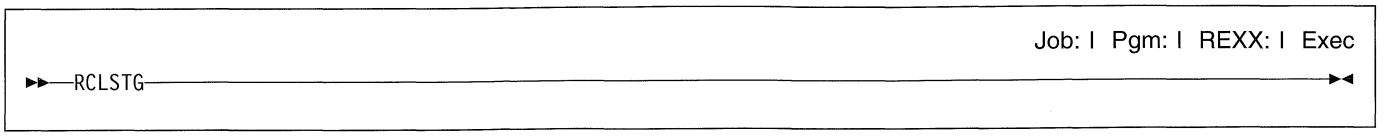
days-to-keep-unused-storage: Specify the number of days to keep unused storage for spooled files. Storage that remains unused for more than the number of specified days is reclaimed.

Example

```
RCLSPLSTG  DAYS(30)
```

This command reclaims all unused storage for spooled files that have remained unused for more than 30 days. When storage has been unused for 1 second over 30 days it is reclaimed because a date and time stamp is placed on the storage area.

RCLSTG (Reclaim Storage) Command



Purpose

The Reclaim Storage (RCLSTG) command is used to perform a general cleanup of auxiliary storage. Use of this command should be considered after an unexpected failure occurs (such as a power or equipment failure) to correct abnormal conditions on permanent objects in auxiliary storage that may be affected by the failure.

The command corrects, where possible, objects that were incompletely updated (such as database files, libraries, and device descriptions) and user profiles containing incorrectly recorded object ownership information. Unusable objects or fragments are deleted.

This command reclaims all objects secured by an authorization list that is damaged or destroyed and assigns the objects to the authorization list QRCLAUTL.

Because the amount of time required to run this command varies with the number of objects in auxiliary storage, the system periodically sends messages to the work station where the command was specified.

The RCLSTG command can also be used to reclaim storage when, during an IPL, not enough storage is available to make the system fully operational. In that case, the system operator can specify the command immediately after receiving the message about insufficient storage.

If very little additional auxiliary storage is available, the system overhead required to run the RCLSTG command may need more than the remaining storage; in that case, the RCLSTG command fails.

Note: The RCLSTG command can be a long-running function, depending on the number and type of objects in the system, the amount of damage to them, the amount of auxiliary storage configured to the system, and the percentage of that storage in use. If database file objects are damaged, the keyed access paths may need to be rebuilt; that operation takes a substantial amount of time. If the RCLSTG command can be run at the user's discretion, the user may want to avoid the operation until the required time can be scheduled.

There are no parameters for this command.

Restrictions:

1. This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.
2. All subsystems must be inactive before the RCLSTG command can be specified. The End System (ENDSYS) or End Subsystem (ENDSBS) command with *ALL specified on the SBS parameter can be used to make the subsystems inactive. You must have job control (*JOBCTL) authority to use the ENDSBS or ENDSYS command.
3. Only permanent objects in auxiliary storage are reclaimed; temporary objects are reclaimed by running a system initial program load (IPL).
4. Before running the RCLSTG command after an IPL, you may need to wait several minutes for the IPL to complete. Use the Work with Active Jobs (WRKACTJOB) command to verify that no jobs are running.
5. This job must be in the controlling subsystem and must be the only job active in the system.

Example

```
RCLSTG
```

This command, specified interactively, locates all system objects created before the last initial program load (IPL). Objects without owners are given default owners, and those that are lost from their specified libraries are inserted into the QRCL library or the default library, or are deleted. Lost objects that are deleted are certain user objects and certain OS/400 system objects that are damaged and not usable. The QRCL library, which is created (when needed) by the RCLSTG command, is a permanent library; but if it contains no objects at the end of the operation because they were all reclaimed, the library is deleted. Objects that are secured by a damaged or destroyed authorization list, or authority holder, are granted authority to the system authorization list QRCLAUTL. Use the Display Authorization List Objects (DSPAUTLOBJ) command to display these objects. When the QRCLAUTL authorization list is needed, it is created by the RCLSTG command.

Additional Considerations

The following statements indicate what happens to objects (including data) that are lost. A lost object can no longer be addressed by the user specifying commands because the system cannot determine which library contains the object.

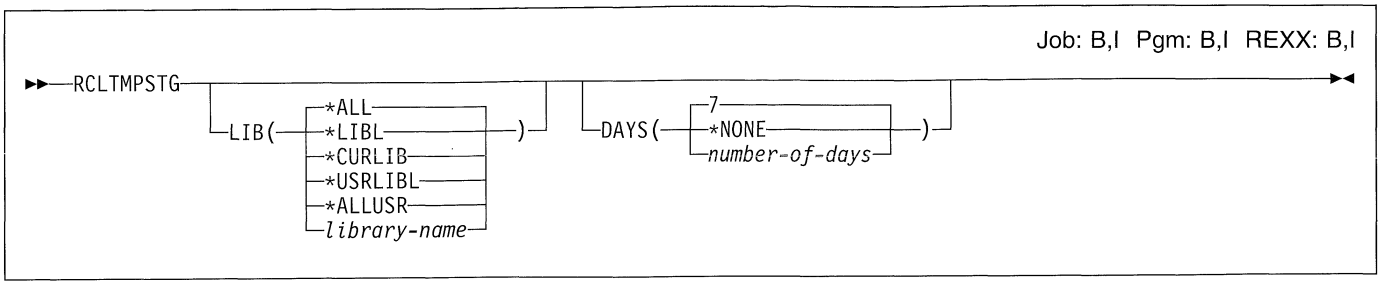
- Objects that cannot be identified as being in any library are placed in the system's reclaim library (QRCL), which (when needed) is automatically created to contain the lost objects. An information message is sent to the

QSYSOPR message queue for each object inserted into QRCL.

Note: For user-created objects remaining in QRCL after storage is reclaimed, the user must either move them to the appropriate library or delete them. It may also be necessary to use the Grant Object Authority (GRTOBJAUT) command to grant specific authority to use the reclaimed objects again.

- Recovered user space (*USRSPC), user index (*USRIDX), and user queue (*USRQ) user domain objects are deleted when neither QRCL nor the special value *ALL are specified on the system value QALWUSRDMN (allow user domain objects in libraries). If these recovered user objects are system domain objects, they are moved to QRCL.
- If more than one lost object has the same name and type, a unique name is assigned to each of those objects, except the first one, and its original name is retained in the object's text description.
- For all lost objects, information, such as save/restore history, programming change status, and text descriptions, may no longer be retrievable, even though the objects themselves can be reclaimed.
- User-created objects whose owners cannot be identified are assigned to the system default owner user profile (QDFTOWN). After storage is reclaimed, the Change Object Owner (CHGOBJOWN) command is used to transfer each object back to its original owner or to a new owner. As an object is transferred to an owner, a message, stating that the object has been transferred and giving the name of the owner to whom it is transferred, is sent to the QSYSOPR message queue.
- Damaged database files for which data still exists are rebuilt as field-level files. These field-level files are identified in the QRCL library in their text descriptions.
- A log of the actions taken by the RCLSTG command (such as objects deleted or renamed, and ownerships transferred) is sent to the QSYSOPR message queue and to the QHST system log. The message CPF8260 is always sent to QHST. The logged objects are displayed by the Display Object Description (DSPOBJD) command for additional information about the damaged or repaired object.

RCLTMPSTG (Reclaim Temporary Storage) Command



Purpose

The Reclaim Temporary Storage (RCLTMPSTG) command removes the temporarily decompressed copies of panel groups, menus, display files, and printer files, thereby freeing up system storage space.

- *Compressed Objects* are objects that consume less storage space than decompressed objects. When a compressed object is used or a compressed program is called, a decompressed version of the object automatically becomes available to the user.
- *Decompressed Objects* are objects that use the system storage space allocated to them and are in a final, ready-to-use state.
- *Temporarily Decompressed Objects* are temporarily decompressed copies of compressed objects. The system allocates storage space for the decompressed objects, which is consumed by the temporary copies until the system or the user determines that the temporary storage space needs to be reclaimed.

  Temporary storage is automatically reclaimed when:

  - The RCLTMPSTG command is run
  - The next initial program load (IPL) is run
  - The object is used often enough to cause the system to permanently decompress it

  When an object is permanently decompressed, the compressed version of the object is destroyed as well as any temporary forms of the object; however, compressed versions remain intact as long as the objects are temporarily decompressed.

Restrictions:

1. This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.
2. The user must have object management authority to the object specified.

Optional Parameters

LIB

Specifies the name of the library in which allocated storage space for temporarily decompressed objects is reclaimed.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

DAYS

Specifies the number of days an object has not been used or changed. If a temporarily decompressed object has not been used or changed for more than the specified number of days, it is reclaimed. If it has been used or changed, it is left temporarily decompressed.

7: Objects that have not been used or changed for more than seven days are reclaimed.

***NONE:** The storage space allocated for a temporarily decompressed object is reclaimed regardless of the

number of days it has not been used or changed, unless the object is in use when this command is run.

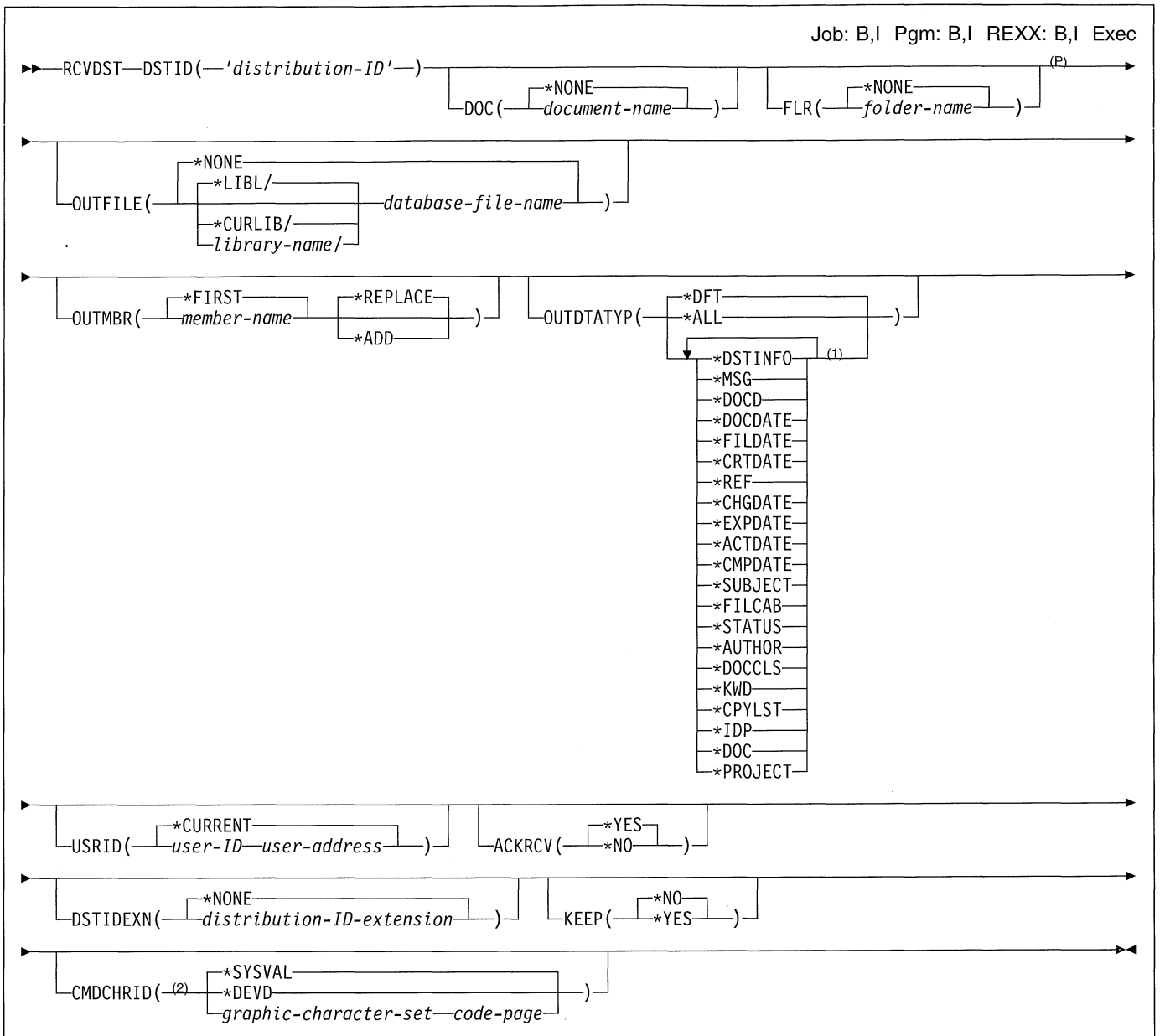
number-of-days: Specify the number of days a temporarily decompressed object can be unused before the temporary space allocated for it is reclaimed. Valid values range from 1 through 366.

Example

```
RCLTMPSTG   LIB(QGPL)
```

This command reclaims the space consumed by all of the temporarily decompressed copies of objects in library QGPL that have not been used or changed in the last 7 days.

RCVDST (Receive Distribution) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 A maximum of 21 repetitions

2 This parameter applies to the DSTID and USRID parameters.

Purpose

The Receive Distribution command allows a user to receive incoming distributions, such as documents. The documents can be placed in folders, in document objects, or in an output file for processing.

Restrictions:

1. Users cannot receive distributions on behalf of another user unless they have been granted permission to work on behalf of that user with the Grant User Permission (GRTUSRPMN) command.
2. The user ID and address must be entered in the system directory.
3. Personal distributions cannot be received if the requester is working on behalf of another user.

Required Parameters

DSTID

Specifies the unique identifier of the distribution. The identifier is assigned to the distribution by the system that creates it. Only incoming distributions can be received. If the identifier represents an outgoing distribution, an error message is returned to the user.

The distribution identifier consists of the sender's address (padded on the right with blanks up to 8 characters), the sender's user ID (padded on the right with blanks up to 8 characters), and a 4-digit zoned sequence number with leading zeros, for example,

```
'NEWYORKbSMITHbbb0204'
```

or

```
MARYLANDMIKEJONE0099
```

Apostrophes are needed if there are blanks or special characters in the distribution identifier. The distribution identifier is specified this way because blank characters are valid in a user ID or address.

Note: If DSTID is the only parameter for which a value is specified, the distribution specified is deleted from the incoming mail log and confirmation of delivery is sent back to the sender, even if ACKRCV(*NO) is specified.

Optional Parameters

DOC

Specifies the document object name under which a distribution is filed when it is received. The document object name that is specified must not exist on the system.

***NONE:** The distribution being received is not filed under a document object name.

document-name: Specify the name of the document object name under which the distribution is filed.

FLR

Specifies the name of the folder that contains the document.

Note: The folder must exist and the current user of this command must have authority to create new documents in the folder.

***NONE:** The distribution being received is not placed in a document object. Specify this value if the distributed document is received directly into a database file for processing.

folder-name: Specify the name of the folder to contain the document object name. A folder name can consist of a series of folder names if the document receiving the distribution is located in a folder contained in another folder. Up to 63 characters can be specified.

OUTFILE

Specifies the name of the database file to which the output is directed. If the output file does not exist, this command creates a database file in the specified library. If the file is created by this function, the text will read "Outfile for RCVDST." The authority for users with no specific authority to use the outfile is *EXCLUDE. More information on defining the format of database files (output files) is in the *Office Services Concepts and Programmer's Guide*.

***NONE:** The output is not directed to a database file.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file that receives the output.

OUTMBR

Specifies the name of the database file member to which the output is directed. If a member already exists, the system uses the second element of this parameter to determine whether the member is cleared before the new records are added. If the member does not exist and a member name is not specified, the system creates a member with the name of the output file specified on the OUTFILE parameter. If an output file member name is specified, but the member does not exist, the system creates it.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter. If the member already exists, there is the option to add new records to the end of the existing member, or to clear the member and then add the new records.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

OUTDTATYP

Specifies which distribution data is written to the database file.

***DFT:** The following record codes are written to the output file:

Record Code	Description
010	Distribution Description
020	Message Text
105	Document Description
800	Document Data

***ALL:** All record formats are written to the output file.

Record Code	Description
010	Distribution Description
020	Message Text
105	Document Description
110	Creation Date
115	Expiration date
120	Document date
125	File date
130	Change date
135	Action due date
140	Completion date
145	Author
150	Copy list
155	Document class
160	File cabinet reference
165	Subject
170	Keyword
175	Reference
180	Status
185	Project
500	Interchange document profile data
800	Document Data

***DSTINFO:** The distribution description record is written to the output file.

***MSG:** The message text record is written to the output file.

***DOCD:** The document description record is written to the output file.

***DOCDATE:** The document date record is written to the output file.

***FILDATE:** The file date record is written to the output file.

***CRTDATE:** The create date record is written to the output file.

***REF:** The reference record is written to the output file.

***CHGDATE:** The change date record is written to the output file.

***EXPDATE:** The expiration date record is written to the output file.

***ACTDATE:** The action due date record is written to the output file.

***CMPDATE:** The completion date record is written to the output file.

***SUBJECT:** The subject records are written to the output file.

***FILCAB:** The filing cabinet reference record is written to the output file.

***STATUS:** The status record is written to the output file.

***AUTHOR:** The author records are written to the output file.

***DOCCLS:** The document class record is written to the output file.

***KWD:** The keyword records are written to the output file.

***CPYLST:** The copy list records are written to the output file.

***IDP:** The complete document record (base sub-profile, application subprofile, and any private sub-profiles) is written to the output file.

***DOC:** The document data records are written to the output file.

***PROJECT:** The project record is written to the output file.

USRID

Specifies the user ID and address of the user for whom the distribution is received. The current user of this command must have the authority to work on behalf of the specified user ID and address.

***CURRENT:** The user profile under which the current job is running is used.

Element 1: User ID

user-ID: Specify the user ID of the user for whom the distribution is received.

Element 2: User Address

user-address: Specify the user address of the user for whom the distribution is received.

ACKRCV

Specifies whether the acknowledgment for confirmation of delivery is sent back to the sender of the distribution.

***YES:** The confirmation of delivery is sent back to the sender.

***NO:** The confirmation of delivery is not sent back to the sender.

DSTIDEXN

Specifies the extension of the distribution identifier specified on the DSTID parameter. This extension uniquely identifies duplicate distributions. This extension is a 2-digit extension that ranges from 01 through 99. For example, if the distribution ID is *NEWYORK SMITH 0204* and two copies of this distribution were sent to a user, then the user has 2 distributions with the same distribution ID. To distinguish the two distributions, an extension is added to each distribution ID and one extension is *NEWYORK SMITH 020401* and the other one is *NEWYORK SMITH 020402*. If there are no duplicates, the extension defaults to 01. These extensions map one to one with the distribution ID specified on the DSTID parameter.

***NONE:** There is no duplicate distribution. This is equivalent to an extension of 01.

distribution-ID-extension: Specify the extension associated with the distribution. This is used to uniquely identify duplicate distribution IDs.

KEEP

Specifies whether this distribution is either deleted from, or kept in, the mail log.

***NO:** When all the information requested has been written to the output file or document object name, the distribution is removed from the user's incoming mail.

***YES:** When all the information requested has been written to the output file or document object name, the distribution is not removed from the user's incoming mail. The incoming distribution is available for another RCVDST request or for processing by another DIA interface, such as OfficeVision/400.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the *Guide to Programming Displays*.

Note: This value translates the DSTID and USRID parameters to the character set and code page of '930 500'. The *Communications: Distribution Services Network Guide* contains the character set and code page table for '930 500'.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values used to create the command parameter.

Element 2: Code Page

code-page: Specify the code page. Valid values range from 1 through 9999.

Examples

Example 1: Receiving Current User Distribution

```
RCVDST   DISTID('SYSTEM1 USERA 0001')
  OUTFILE(MYLIB/MYFILE)
  OUTMBR(MYMBR *ADD)  OUTDTATYP(*ALL)  CMDCHRID(*DEVD)
```

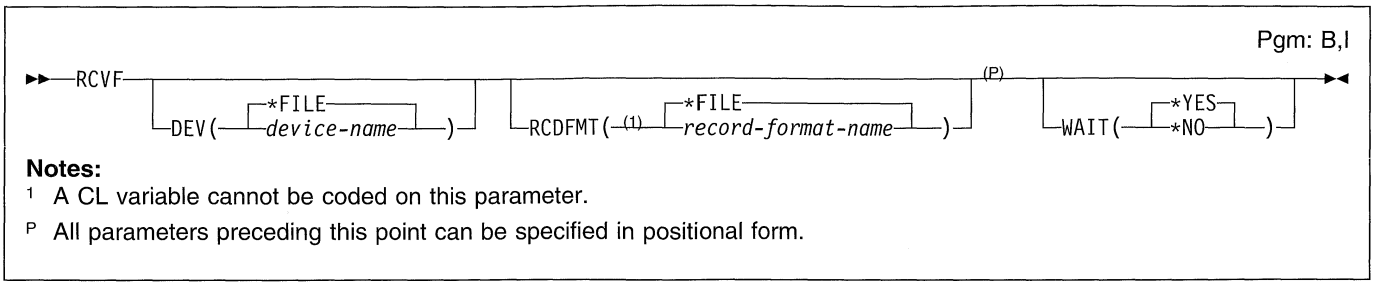
This command receives the current user distribution into output file MYFILE located in library MYLIB. The distribution is added to member MYMBR. All output file information is added to the output file MYFILE.

Example 2: Receiving Distribution Sent to a User

```
RCVDST   DSTID('BAKERbRCH38Pb0019')  DSTINDEXN(01)
  OUTFILE(JOWLIB/DOCUMENTS)  USRID(*CURRENT)
```

This command receives a distribution that was sent to a user. It is copied into the first member in a database file called DOCUMENTS in a library called JOWLIB.

RCVF (Receive File) Command



Purpose

The Receive File (RCVF) command is used within a CL program to receive data from a display device or database file. The command reads a record from the file and puts the data from the record into one or more CL variables. These CL variables were automatically declared in the program when the CL source program was compiled and a Declare File (DCLF) command was processed as part of the source. There is one CL variable for each field in the record format used to receive the data. The data that is entered by a user at the display or is contained in the input record is copied into CL variables in the program by the RCVF command, where it is processed by the program.

Only one record format, of those specified in the DCLF command, can be specified in each RCVF command. If the file has not been opened by a previous RCVF, SNDRCVF, or SNDF command, it is opened by this command. If the file has been previously closed due to an end-of-file condition on a previous RCVF command, an error occurs. The file specified in this command can be overridden if the override command is entered before the file is opened. If the file specified in the DCLF command was a display file when the program was compiled, the file may only be overridden to another display file. If the file was a database file, the file may only be overridden to another database file that has a single record format. However, care should be taken that the fields in the overriding record format correspond to the CL variables declared in the program.

Optional Parameters

DEV

Specifies the name of the display device from which data is being received. If a CL variable name is used in this parameter, only one RCVF command is needed in the program to receive data from several devices. (The variable specifying the device name can be changed while repeatedly running the same command.) This parameter may be specified only if the file is a display device file.

***FILE:** The user's data is received from the device associated with the device file (the device file that was declared in the FILE parameter of the DCLF command). If more than one device name is specified in the device file, *FILE cannot be specified.

device-name: Specify the name of the device or the name of the CL variable that contains the name of the device from which the user's data is being received.

RCDFMT

Specifies the name of the record format that is used to receive data from the file. The format contains all the fields in the record. This parameter must be coded with a record format name if there is more than one record format in the device file. If the file is a database file, the specified record format is used to map the data from the record into the CL variables. The actual record format name in the file at run time may be different. RCVF ignores the INVITE DDS keyword.

***FILE:** There is only one record format in the device file; that is the format in which the data is being received. If more than one record format is specified in the device file, *FILE cannot be specified.

record-format-name: Specify the name of the record format in which the data records from the display device are being received. A CL variable cannot be used to specify the record format name.

WAIT

Specifies whether the CL program waits for the data being received from the user's device or continues processing the commands that follow this RCVF command. If WAIT(*NO) is specified, the program must issue a WAIT command later in the program to complete the input operation. This parameter may be specified only if the file is a display device file.

***YES:** The program waits until the input operation from the device is completed; the next command is not processed until then.

***NO:** The program does not wait for the input data; commands continue running until a WAIT command is reached later in the program.

Examples

```
DCLF    FILE(MENU1)
  •
  •
  •
RCVF
```

The CL program receives data from the user through the file named MENU1. The program waits for the user data before it continues processing.

```
DCLF    FILE(SCREENX)  RCDFMT(R1 R2)
  •
RCVF    DEV(DISPLAY2)  RCDFMT(R1)
```

The CL program receives data from the user at the display station named DISPLAY2. The data is received in the record format named R1 in the device file named SCREENX. The program waits for the user data before it continues processing.

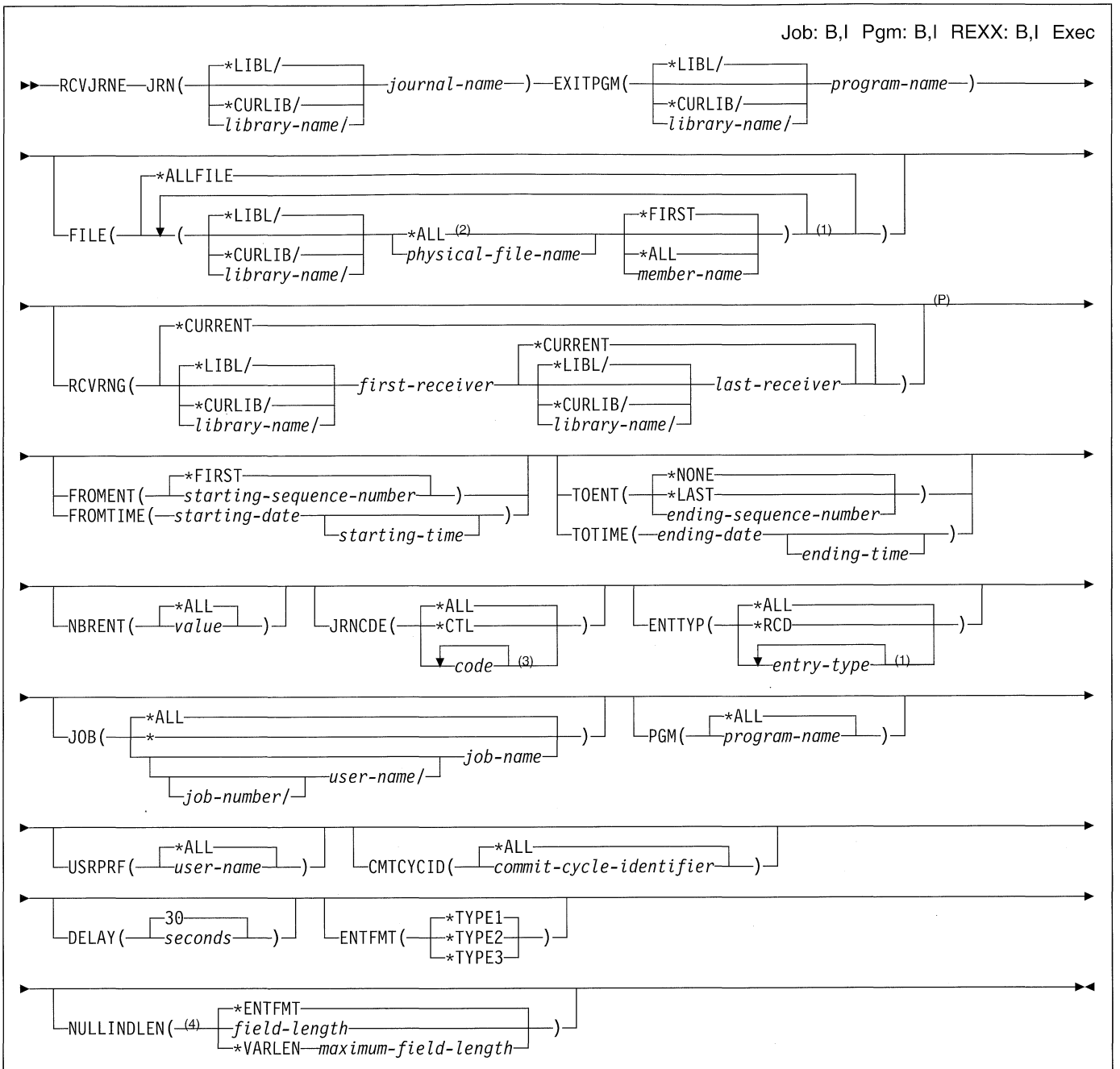
```
DCLF    FILE(INPUT)
  •
RCVF
MONMSG    CPF0864  EXEC(GOTO EOF)
```

The CL program receives a record sequentially from the database file named INPUT. The program monitors for the end-of-file exception CPF0864 and goes to label EOF when the message is received.

```
DCLF    FILE(MSCREEN)  RCDFMT(MIN1 MIN2 MIN3)
  •
  •
  •
RCVF    DEV(&DNAME)  RCDFMT(MIN2)  WAIT(*NO)
WAIT    DEV(&DNAME)
```

The CL program receives user data from several devices one at a time by way of the device file named MSCREEN. The program receives data from the device named in the variable &DNAME using the record format MIN2, but it does not wait for the data to come in. The same RCVF command is used to receive data from several devices; because the CL variable &DNAME is used, only the device name in the DEV parameter must be changed each time the command is run. A WAIT command for each device must be issued later in the program because the WAIT command actually receives the data. Both the RCVF and the WAIT commands may be processed for each device (one at a time) to send data to the program. If a user response is delayed, the commands can be processed as many times as necessary until the user responds with the data or a End Receive (ENDRCV) command cancels the request.

RCVJRNE (Receive Journal Entry) Command



Notes:

1 A maximum of 50 repetitions

2 If *ALL is specified instead of physical-file-name, the format must be library-name/*ALL.

P All parameters preceding this point can be specified in positional form.

3 A maximum of 10 repetitions.

4 The NULLINDLEN parameter can be specified only if ENTFMT(*TYPE 3) is specified.

Purpose

The Receive Journal Entry (RCVJRNE) command allows a specified user exit program to continuously receive journal entries one at a time. This program can be set up, for example, to write the entries either (1) to an ICF file, supplying updates to a file on a backup system, or (2) on a tape, imitating a journal-to-tape function. The journal entries received can be used to update the database files being journaled to minimize the loss of data in the event of a disk failure, and to update database files on a backup system in case of a system failure on the primary system.

The value specified on the ENTFMT parameter determines the format of the journal entries passed to the exit program.

Restrictions:

1. If the sequence number is reset in the range of receivers specified, the first occurrence of the FROMENT or TOENT parameter is used if either is specified.
2. If more than one selection value is specified, a journal entry must satisfy all of the selection values to be received. Selection values are FILE, JRNCDE, ENTTYP, JOB, PGM, and CMTCYCID. If none of these parameters are specified, all entries are received as specified in the values given for the RCVRNG, FROMENT, TOENT, FROMTIME, TOTIME, and NBRENT parameters.

Required Parameters

JRN

Specifies the qualified name of the journal where the journal entries are to be received.

The name of the journal can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

journal-name: Specify the name of the journal where the journal entries are to be received.

EXITPGM

Specifies the qualified name of a user-written exit program that is given control to receive each journal entry passed from the command.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

When the program is called, two parameters are passed to it at a time. The journal entry is passed in the first parameter. If the length of the parameter defined by the program is smaller than the length of the journal entry, the journal entry passed to the program is truncated. If the length of the parameter defined by the program is greater than the length of the journal entry, the parameter positions beyond the length of the journal entry contain nonessential information. The user's program should not specifically refer to data in the positions beyond the length of the journal entry. The maximum parameter length is 9999 for CL and RPG, and 32767 for all other languages. The format of the information in each journal entry is shown in the ENTFMT parameter.

The second parameter is a character variable with LEN(1) that can have one of three values. The first value, 9, is set by the exit program to signal the RCVJRNE function to end. The Send Journal Entry (SNDJRNE) command can be used to send an entry signaling the exit program to set the end value.

The second value, 0, is set by the RCVJRNE function to inform the exit program that no entry is being passed because there are no new entries in the journal. This allows the exit program to clear its buffers, ensuring a more current save operation. When control is returned to the system module, a delay takes place as determined by the DELAY parameter, regardless of whether new entries are reached. The 0 value is passed before each consecutive delay to allow the program to send an end value (9) after too many delays.

The third value, 1, is set by the RCVJRNE function when an entry is passed to the exit program.

program-name: Specify the name of the exit program that controls the reception of each journal entry passed from the command.

Optional Parameters

FILE

Specifies a maximum of 50 qualified file names whose journal entries are received. The FILE parameter also specifies the name of the file member whose journal entries are to be received.

If the specified physical file member currently exists on the system, all journal entries in the specified receiver range for that physical file member are received. If the specified physical file member does not exist on the system, journal entries in the specified receiver range for any physical file member of that name that previously existed on the system are received.

***ALLFILE:** The search for the entries received is not limited to a particular file.

Element 1: Physical File Name

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

***ALL:** The search for the entries received is for all files in the assigned library that are currently being journaled. When FILE(*ALL) is specified, the file library qualifier cannot be *LIBL.

physical-file-name: Specify the name of the physical database file for which a journal entry is received.

Element 2: Member Name

***FIRST:** Entries for the first member in the file are received.

***ALL:** Entries for currently existing members of the file are converted for output.

member-name: Specify the name of the member for which journal entries are received.

If the specified physical file does not exist on the system, specify either *ALL or a specific file member name.

RCVRNG

Specifies the first (beginning) and last (ending) journal receivers used in the search for the journal entries that are received. The system starts the search with the first journal receiver (as specified by the first value) and proceeds through the receiver chain until the last receiver (as specified by the last value) is processed.

If dual receivers (receivers attached and detached in pairs) are used at any time, the system uses the first of the paired receivers when chaining through the receivers. The Work with Journal Attributes (WRKJRNA) command can be used to display the order of the receivers in the receiver chain.

If a problem is found in the receiver chain (such as damaged or not-found receivers) before the search operation begins, the system tries to use the second of the dual receivers. If these receivers also are damaged or not found, the operation ends.

***CURRENT:** The journal entries in the currently attached receiver are received.

Element 1: First Receiver

The name of the journal receiver can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

first-receiver: Specify the name of the first journal receiver that contains the journal entries to be received.

Element 2: Last Receiver

***CURRENT:** The search for journal entries continues for all journal receivers in the chain that began with the receiver specified by the first parameter value and continues through the currently attached journal receiver.

The name of the journal receiver can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

last-receiver: Specify the qualified name of the last journal receiver that contains journal entries to be received. If the end of the receiver chain is reached before a receiver with this name is found, an error message is sent and no journal entries are received.

Note: The maximum number of receivers in the range is 256. If more receivers than this maximum are specified, an exception is signaled, and no journal entries are received.

FROMENT

Specifies the first journal entry considered for reception.

***FIRST:** The first journal entry in the journal receiver range specified is the first entry considered for reception.

starting-sequence-number: Specify the sequence number of the first journal entry considered for reception.

FROMTIME

Specifies the date and time of the first journal entry considered for reception. The journal entry with the specified date and time or the next later journal entry is the starting point for reception of journal entries.

Element 1: Starting Date

starting-date: Specify the date of the first journal entry considered for reception. The format of the date must be as defined by the job attributes DATFMT and, if separators are used, DATSEP.

Element 2: Starting Time

starting-time: Specify the time of the first journal entry considered for reception. The time is specified in 24-hour format with or without a time separator as follows:

- With a time separator, specify a string of 5 or 8 digits where the time separator separates the hours, minutes, and seconds. If this command is entered from the command line, the string must be enclosed in apostrophes. If a time separator other than the separator specified for your job is used, this command fails.
- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where **hh** = hours, **mm** = minutes, and **ss** = seconds. Valid values for **hh** range from 00 through 23. Valid values for **mm** and **ss** range from 00 through 59.

TOENT

Specifies the last journal entry considered for reception.

***NONE:** No journal entry is specified. Journal entries are passed to the exit program until the command is canceled (cancel request or cancel job) or until an end return code (9) is set by the exit program. If there are no more entries to pass, the RCVJRNE command waits the number of seconds indicated on the DELAY parameter before trying to find more entries to pass.

***LAST:** The last journal entry in the journal receiver range specified is the last journal entry considered for reception.

ending-sequence-number: Specify the sequence number of the final journal entry considered for reception.

Note: The values specified for the from and to parameters can be the same. For example, FROMENT(234) and TOENT(234) can be specified.

TOTIME

Specifies the date and time of the last journal entry considered for reception. The first journal entry at or before the specified date and time is the last journal entry considered for reception.

Element 1: Ending Date

ending-date: Specify the date of the last journal entry considered for reception. The format of the date must be as defined by the job attributes DATFMT and, if separators are used, DATSEP.

Element 2: Ending Time

ending-time: Specify the time of the last journal entry considered for reception. See the FROMTIME parameter for a description of time formats.

NBRENT

Specifies the total number of journal entries to receive.

***ALL:** All journal entries that are for the specified journal receivers and that satisfy the selection values are received.

value: Specify the maximum number of journal entries to receive. If the specified journal entry identified by the TOENT or TOTIME parameter is reached before the value specified for NBRENT is met, the command ends normally.

JRNCDE

Specifies whether the entries being considered are limited to the journal entries that contain the specified journal code.

***ALL:** The journal entries received are not limited to those containing a specified code.

***CTL:** The journal entries received are those written to control the journal functions. These journal entries have codes 'J' and 'F'.

code: Specify the journal code that limits the journal entries to be received. Only journal entries that contain the identified journal code are considered for reception. A list of journal codes that can be specified is in the *Advanced Backup and Recovery Guide*.

ENTTYP

Specifies that the entries received are limited to journal entries that contain the specified entry type.

***ALL:** The journal entries to be received are not limited to a specified entry type.

***RCD:** The journal entries received are limited to those written for record level operations. These are entry types: BR, DL, DR, PT, PX, UB, UP, and UR.

entry-type: Specify the entry type that limits the journal entries received. Only journal entries that contain the specified entry type are received. Up to 50 valid entry types can be specified. A list of valid entry types that can be received is in the *Advanced Backup and Recovery Guide*.

JOB

Specifies whether the journal entries received are limited to those for a specified job. If the fully qualified name of the job is not specified, all journal entries that contain the simple job name are considered for reception.

A job identifier is a special value or a qualified name with up to three elements. For example:

```
*ALL
*
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ALL:** The journal entries received are not limited to a specified job.

***:** The journal entries received are limited to those for the current job.

job-name: Specify the name of the job whose journal entries are considered for reception.

user-name: Specify the name of the user of the job whose journal entries are considered for reception.

job-number: Specify the number of the job whose journal entries are considered for reception.

PGM

Specifies whether the journal entries received are limited to the those created for a specified program.

***ALL:** The journal entries received are not limited to those created by a specified program.

program-name: Specify the name of the program whose journal entries are considered for reception. Only journal entries for this program are considered for reception.

USRPRF

Specifies that the journal entries to be received are limited to the journal entries for a specified user profile name.

***ALL:** The journal entries received are not limited to entries for a specified user profile.

user-name: Specify the name of the user profile whose journal entries are considered for reception. Only journal entries for this user profile are considered for reception.

CMTCYCID

Specifies that the journal entries are limited to the journal entries that contain the specified commit cycle identifier.

***ALL:** The journal entries received are not limited to a specified commit cycle identifier.

commit-cycle-identifier: Specify the commit cycle identifier that limits the journal entries received. Only journal entries that contain the commit cycle ID are considered for reception.

DELAY

Specifies the number of seconds that the command waits for a new journal entry to arrive if the last entry has already been received. After the last entry in the journal is received and passed to the exit program, the command tries to receive the next entry. If no new journal entry exists, the exit program is passed a value of 0 in the second parameter. When control is returned from the exit program, the command delays the specified number of seconds. If the entries have arrived while the exit program is not in operation, they are not received until the time delay is completed.

Restrictions: The DELAY parameter is valid only when TOENT(*NONE) is specified, and the ending receiver specified on the RCVRNG parameter identifies a currently attached journal receiver.

30: The command waits 30 seconds before trying to find a new journal entry on the journal.

seconds: Specify the number of seconds that the command waits for the arrival of a new journal entry on the journal. Valid values range from 1 through 99999.

ENTFMT

Specifies the format of the journal entries being received.

***TYPE1:** The received journal entries do not include the user profile field, the system name field, or the null value indicators. The format of the information in each journal entry is shown below:

*Table 65. *TYPE1 Journal Entry Format*

	Length	From	To
Entry Length	5	1	5
Sequence Number	10	6	15
Journal Code	1	16	16
Journal Entry Type	2	17	18
Date	6	19	24
Time	6	25	30
Job Name	10	31	40
User Name	10	41	50
Job Number	6	51	56
Program Name	10	57	66
Object Name	10	67	76
Object Library	10	77	86
Member Name	10	87	96
Count/RRN	10	97	106
Flag	1	107	107
Commit Cycle ID	10	108	117
Reserved	8	118	125
Entry-specific Data	N[1]	126	N+125

Note:

1 The length of the entry-specific data field varies from entry to entry. It is long enough to accommodate all the entry-specific data in each received journal entry.

***TYPE2:** The entries received include the information returned when ENTFMT(*TYPE1) is specified, the user profile field, which gives the name of the user who caused the logging of the received journal entries, and the name of the system on which the entry was sent.

The format for *TYPE2 journal entries is shown below.

*Table 66 (Page 1 of 2). *TYPE2 Journal Entry Format*

	Length	From	To
Entry Length	5	1	5
Sequence Number	10	6	15
Journal Code	1	16	16
Journal Entry Type	2	17	18

Table 66 (Page 2 of 2). *TYPE2 Journal Entry Format			
	Length	**From**	**To**
Date	6	19	24
Time	6	25	30
Job Name	10	31	40
User Name	10	41	50
Job Number	6	51	56
Program Name	10	57	66
Object Name	10	67	76
Object Library	10	77	86
Member Name	10	87	96
Count/RRN	10	97	106
Flag	1	107	107
Commit Cycle ID	10	108	117
User Profile	10	118	127
System Name	8	128	135
Reserved	20	136	155
Entry-specific Data	N[1]	156	N + 155

Note:

1 The length of the entry-specific data field varies from entry to entry. It is long enough to accommodate all the entry-specific data in each received journal entry.

***TYPE3:** The journal entries received include the information returned when ENTFMT(*TYPE2) is specified, and contain null value indicators. The format of the received entries depends on the value specified on the NULLINDLEN parameter. The tables in the NULLINDLEN parameter description show the three formats for *TYPE3.

NULLINDLEN

Specifies the length, in bytes, used for the null value indicators portion of the journal entry received by the user. This parameter is valid only if ENTFMT(*TYPE3) is specified.

***ENTFMT:** All null value indicators are received for each journal entry. The format for this value is shown in Table 67.

Note: The number of null value indicators, as well as the length of the entry-specific data can vary from entry to entry. In Table 67, the number of null value indicators is designated by the variable 'M' and the length of the entry-specific data is designated by the variable 'N'.

Table 67. NULLINDLEN(*ENTFMT) Journal Entry Format			
Field Name	**Length**	**From**	**To**
Entry Length	5	1	5
Sequence Number	10	6	15
Journal Code	1	16	16

Table 67. NULLINDLEN(*ENTFMT) Journal Entry Format			
Field Name	**Length**	**From**	**To**
Journal Entry Type	2	17	18
Timestamp	26	19	44
Job Name	10	45	54
User Name	10	55	64
Job Number	6	65	70
Program Name	10	71	80
Object Name	10	81	90
Object Library	10	91	100
Member Name	10	101	110
Count/RRN	10	111	120
Flag	1	121	121
Commit Cycle ID	10	122	131
User Profile	10	132	141
System Name	8	142	149
Number of Null Value Indicators[1]	5	150	154
Null Value Indicators	M	155	154+M
Length of Entry-Specific Data[2]	5	155+M	159+M
Entry-Specific Data	N	160+M	159+M+N

Notes:

1 This field contains the number of null value indicators (in decimal digits) in the received journal entry.

2 This field contains the length of the entry-specific data (in decimal digits) in the received journal entry.

field-length: Specify the field length for the null value indicators portion of the received journal entry. Valid values range from 1 to 8000 characters. The format of the received journal entry is shown in Table 68.

Note: The number of entry-specific data can vary from entry to entry. In Table 68 and Table 69, the length of the entry-specific data is designated by the variable 'M'.

Table 68 (Page 1 of 2). NULLINDLEN(field-length) Journal Entry Format			
Field Name	**Length**	**From**	**To**
Entry Length	5	1	5
Sequence Number	10	6	15
Journal Code	1	16	16
Journal Entry Type	2	17	18
Timestamp	26	19	44
Job Name	10	45	54
User Name	10	55	64
Job Number	6	65	70
Program Name	10	71	80

Table 68 (Page 2 of 2). NULLINDLEN(field-length) Journal Entry Format

Field Name	Length	From	To
Object Name	10	81	90
Object Library	10	91	100
Member Name	10	101	110
Count/RRN	10	111	120
Flag	1	121	121
Commit Cycle ID	10	122	131
User Profile	10	132	141
System Name	8	142	149
Null Value Indicators	*field length*[1]	150	149+ *field length*
Entry-Specific Data	M[2]	150+ *field length*	149+ M+ *field length*

Notes:

1 The length of the null value indicators field is the length specified on the NULLINDLEN parameter.

2 The length of the entry-specific data field varies from entry to entry. It is long enough to accommodate all the entry-specific data in each received journal entry.

If the journal entry being passed to the exit program has fewer null value indicators than the length specified on the NULLINDLEN parameter, the trailing bytes are set to 'F0'X. Conversely, if a journal entry received has more null value indicators than the specified field length and truncation will result in the loss of a 'F1'X indicator value, the RCVJRNE request is ended.

Element 1: Variable-Length Field

***VARLEN:** The null value indicators field is a variable-length field. The received journal entry has the format specified in Table 69.

Element 2: Maximum Field Length

The maximum length of the null value indicators field.

maximum-field-length: Specify the maximum number of null value indicators to be included in each received journal entry. Valid values range from 1 to 8000. If a journal entry received has more null value indicators than the specified field length and truncation will result in the loss of a 'F1'X indicator value, the RCVJRNE request is ended.

*Table 69. NULLINDLEN(*VARLEN field-length) Journal Entry Format*

Field Name	Length	From	To
Entry Length	5	1	5
Sequence Number	10	6	15
Journal Code	1	16	16
Journal Entry Type	2	17	18
Timestamp	26	19	44
Job Name	10	45	54
User Name	10	55	64
Job Number	6	65	70
Program Name	10	71	80
Object Name	10	81	90
Object Library	10	91	100
Member Name	10	101	110
Count/RRN	10	111	120
Flag	1	121	121
Commit Cycle ID	10	122	131
User Profile	10	132	141
System Name	8	142	149
Number of Null Value Indicators[1]	2	150	151
Null Value Indicators	*field length*	152	151+ *field length*
Length of Entry -Specific Data[2]	2	152+ *field length*	153+ *field length*
Entry-Specific Data	M	154+ *field length*	153+ M+ *field length*

Notes:

1 This field contains the number of null value indicators (in binary digits) in the received journal entry.

2 This field contains the length of the entry-specific data (in binary digits) in the received journal entry.

Examples

Example 1: Receiving Journal Entries

```
RCVJRNE   JRN(APPLIB/JRN1)  EXITPGM(MYLIB/RCVPGM)
  FILE(APPLIB/FILE3)  TOENT(*LAST)
  ENTFMT(*TYPE3)  NULLINDLEN(*ENTFMT)
```

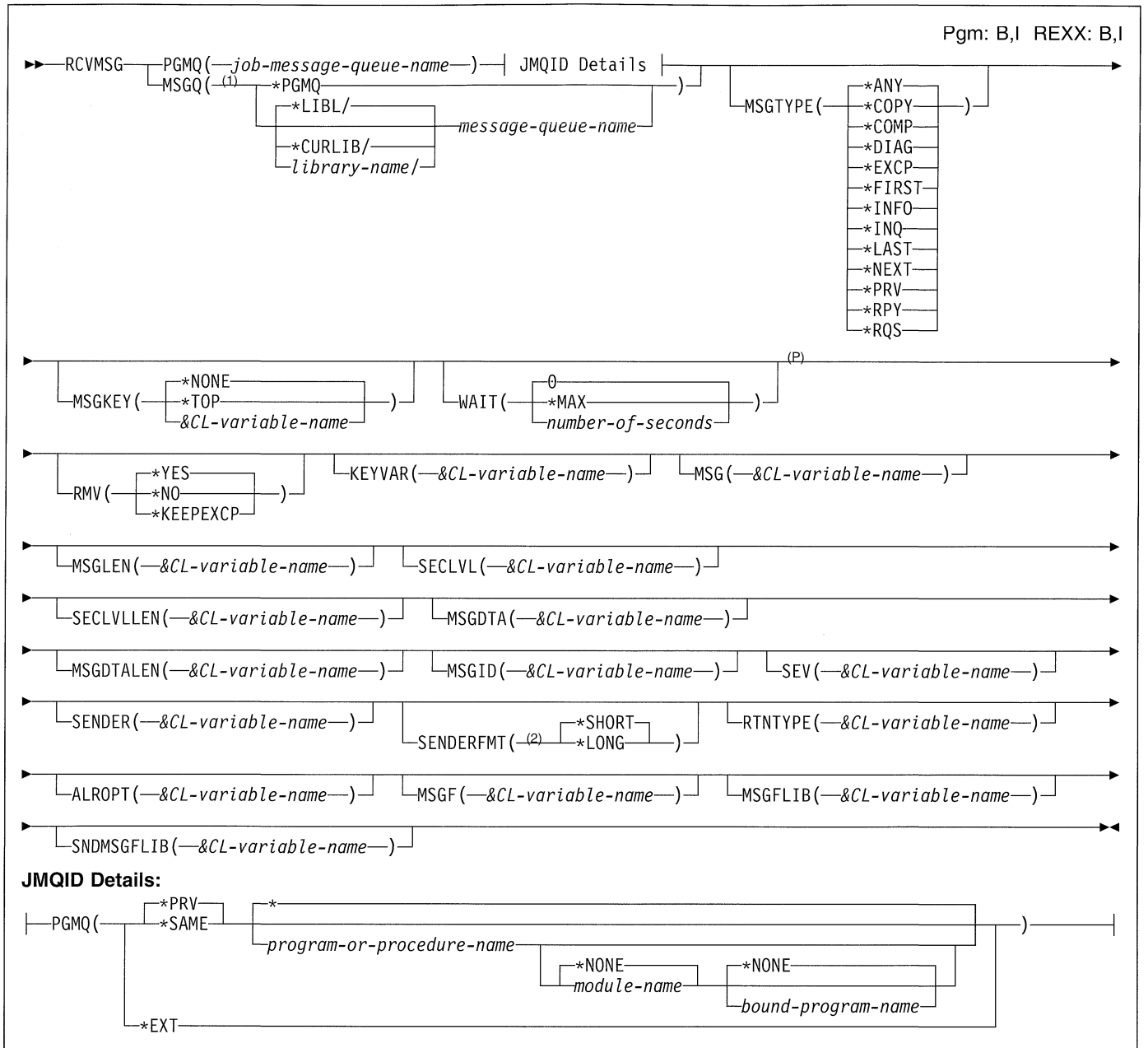
This command receives journal entries from the journal receiver currently attached to journal JRN1 in library APPLIB, and passes them one at a time to program RCVPGM in library MYLIB. Only entries with file-level information for the first member of file FILE3 in library APPLIB are received. The format of each entry passed to the exit program is shown in Table 67.

Example 2: Receiving Journal Entries

```
RCVJRNE   JRN(JRNLIB/MYJRN)  EXITPGM(RCVLIB/PGMA)
  FILE(FILELIB/PFILEB MBRONE)  TOENT(*LAST)
  ENTFMT(*TYPE3)  NULLINDLEN(*VARLEN 30)
```

This command receives journal entries with file-level information for member MBRONE of file PFILEB in library FILELIB from the journal receiver currently attached to journal MYJRN in library JRNLIB and sends them one at a time to program PGMA in library RCVLIB. The format of each entry passed to the exit program is shown in Table 69. The null value indicators portion of each received entry is 30 characters in length.

RCVMSG (Receive Message) Command



Notes:

1 When a value is specified for the PGMQ parameter, MSGQ(*PGMQ) is allowed.

2 This parameter is valid only when the SENDER parameter is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Receive Message (RCVMSG) command is used by a program to receive a message that was previously sent to a message queue.

The RCVMSG command receives messages from a job message queue (a message queue associated with a call stack entry or the external message queue (*EXT)), or from a named message queue. The program can receive a message from a message queue associated with its own call stack entry or from a message queue associated with another call stack entry.

This command copies a message received in the specified message queue into control language (CL) variables within the program. The message and its attributes are copied into

the CL variables specified by the parameters KEYVAR through SNDMSGFLIB.

You can specify the message being received by indicating the message type, the reference key of the message, or both. The program receiving the message can also specify, on the RCVMSG command, whether a message is removed from the message queue or left there as an old message. If the specified message queue is not allocated to the job in which this command is entered, or to any other job, the message queue is implicitly allocated by this command for the duration of the command's processing.

If a message of the specified type has not been received by the queue, the requesting program can either wait for a message to arrive or continue with other processing. This allows a set of message queues to be polled.

If the message received is an unhandled exception message, the program can specify whether this command should handle the exception. An unhandled exception message is an escape, status, or notify message that has been sent to an Integrated Language Environment (ILE) procedure. When this command is run, the ILE procedure has not yet taken action to tell the system that the exception is handled. One action the ILE procedure can take is to call a CL program that receives the message using this command. More information on actions that can be taken is in *ILE* Concepts*.

Restriction: This command is valid only in compiled CL programs.

Optional Parameters

PGMQ

Specifies the call stack entry message queue from which a message is received. The call stack entry message queue can be the *EXT queue or it can be a message queue that is associated with a call stack entry for a program or an ILE procedure.

Element 1: Relationship

Element 1 of this parameter specifies whether the message queue is associated with the program or procedure identified by Element 2, or if it is associated with the caller of the program or procedure.

***PRV:** The message is received from the message queue of the program or procedure that called the program or procedure identified by Element 2.

***SAME:** The message is received from the message queue of the program or procedure identified by Element 2.

Element 2: Program or Qualified Procedure

Element 2 of this parameter has three items. Item 1 specifies the program or procedure of the job message queue. Items 2 and 3 specify the module name and the bound program name, which can be used to qualify the procedure name.

Item 1: Program or Procedure Name

***:** Identifies the program running the RCVMSG command.

program-or-procedure-name: Specify the name of the program or procedure used.

If Item 1 identifies a program, the name specified can be a maximum of 10 characters. If Item 1 identifies a procedure, the name specified can be a maximum of 256 characters. The system begins its search for the specified program or procedure name with the most recently called program or procedure.

The procedure name alone may not identify the correct procedure. Several different procedures with the same name can run in a job. To further identify a procedure, the name specified can be qualified by a module name, or by both a module name and a bound program name.

Item 2: Module Name

The module name qualifier identifies the module into which the procedure was compiled.

***NONE:** No module name is specified.

module-name: Specify the module name to be used as a qualifier for the specified procedure name (modules are associated only with procedures). The module name can be a maximum of 10 characters.

If a module name is not specified but a bound program name is, *NONE must be specified in the module name position.

Item 3: Bound Program Name

The bound program name qualifier identifies the program to which the procedure was bound.

***NONE:** No bound program name is specified.

bound-program-name: Specify the bound program name to be used as a qualifier for the specified procedure name (and module name if specified). The bound program name can be a maximum of 10 characters.

Single Value

***EXT:** The message is received from the external message queue of the job. The external message queue is used to communicate with the external requester of the job, such as a display station user.

MSGQ

Specifies the qualified name of the message queue (not a program message queue) from which a message is received. If MSGQ is specified, the PGMQ parameter cannot be specified.

***PGMQ:** The job message queue specified in the PGMQ parameter is the only queue from which a message is received.

The possible library values are:

***LIBL:** The library list is used to locate the message queue.

***CURLIB:** The current library for the job is used to locate the message queue. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library where the message queue is located.

message-queue-name: Specify the name of the message queue from which a message is received.

MSGTYPE

Specifies the type of message received by this program. The message types *EXCP, and *RQS exists only on job message queues and therefore can be received only from a job message queue. For the coding relationships between the MSGTYPE and MSGKEY parameters, see Coding Relationships in the MSGKEY parameter description.

***ANY:** Any type of message (except a sender's copy) is received. To receive a sender's message, MSGTYPE(*COPY) must be specified.

***COPY:** A copy of an inquiry message that was previously sent is received by this program. The message queue specified for the PGMQ or MSGQ parameters must be the same queue that was specified for the RPYMSGQ parameter when the inquiry message was sent.

***COMP:** A completion message is received. It indicates the status of the work that this program requested of another program.

***DIAG:** A diagnostic message is received. It provides information about errors detected by another program in the input sent by this program or errors that occurred when the requested function was being processed by the other program.

***EXCP:** An exception message is received. Exception messages (escape, notify, status) are received by the program in last-in first-out (LIFO) order. The receiving program can monitor for exception messages by using the MONMSG command.

Note: Non-exception messages are received in first-in first-out (FIFO) order.

If an exception message is received from a message queue for a procedure, the related exception may not be handled at the time the RCVMSG command is run. The RMV parameter can be used to specify whether the exception is to be handled by the RCVMSG command.

***FIRST:** The first message currently on the specified message queue is received.

***INFO:** An information only message is received.

***INQ:** An inquiry message is received.

***LAST:** The last message currently on the specified message queue is received.

***NEXT:** The message that follows the one specified in the MSGKEY parameter is received. If there is not another message available, blanks are returned in all CL variables.

When a message is received from a message queue associated with a call stack entry, *NEXT works only for one call stack entry. *NEXT cannot be used to receive messages for multiple call stack entries of the same program.

***PRV:** The message previous to the message indicated by the MSGKEY parameter is received.

***RPY:** A reply message is received. This program has sent an inquiry message to a message queue and expects a reply.

***RQS:** A request message is received. The message specifies a request for a function. The receiving program can then perform the function requested.

MSGKEY

Specifies the message reference key of the message that is received.

***NONE:** No message reference key is specified.

***TOP:** The top of the message queue is used. *TOP can be used only when MSGTYPE(*NEXT) is specified. It causes the first message on the message queue to be received. For program message queues, this is the message following the last request message that was received, if any.

&CL-variable-name: Specify the name of the control language (CL) variable that contains the message reference key of the message received by this program. This key is assigned by the system and cannot be shown. The variable must be a character variable having a length of 4 characters.

Coding Relationships: The MSGTYPE and MSGKEY parameters can be used separately in the RCVMSG command, or together.

- If neither MSGTYPE nor MSGKEY is specified, MSGTYPE(*ANY) is assumed and the first *new* message in the queue is received; that is, the messages are received in FIFO (first-in, first-out) order.
- If one of the message types specified on the MSGTYPE parameter is *COMP, *DIAG, *INFO, *INQ, *RPY, *COPY, or *RQS, a new message of the specified type is received in FIFO order. If the type is *EXCP, new messages are received in LIFO (last-in, first-out) order.
- If only MSGKEY is specified with a CL variable name and the message queue contains a message with the specified message reference key, that message is received. If the MSGKEY specified is for a sender's copy message, the reply to the message is received if it is available. If the reply is not available, blanks will be returned in all CL variables. If a message is requested by key and the message is not available, an escape message is sent to the requesting program.

- If MSGTYPE(*COPY) and MSGKEY (&CL-variable-name) are specified, the sender's copy of an inquiry message is received.
- If both MSGTYPE and MSGKEY (&CL-variable-name) are specified and the message queue has a message of that type, the message is received by the program. If the reference key is correct and the message type is not, then an error message is sent to the program.
- If MSGTYPE(*NEXT) is specified, MSGKEY must be specified. The message following the message with the specified reference key is received. If MSGKEY(*TOP) is specified with MSGTYPE(*NEXT), the first message on the message queue is received. For call message queues, this is the first message following the last request message received.

If MSGTYPE(*RPY) is specified with a MSGKEY value that refers to either a sender's copy or an inquiry message, any reply to the sender's copy or inquiry message is returned. If there is no reply to that sender's copy or inquiry message, blanks are returned. When the MSGKEY value refers to an inquiry message, the WAIT parameter is ignored (this implies WAIT(0), which is the default value for the WAIT parameter).

If MSGTYPE(*ANY) is specified with a KEYVAR variable and the first message type found is a reply message, the KEYVAR variable returns the message reference key of the sender's copy message. Similarly, if MSGTYPE(*RPY) is specified with a KEYVAR variable, the message reference key of the sender's copy message is returned.

WAIT

Specifies the length of time (in seconds) that the program waits for a message of the specified type to arrive in the message queue if it is not there when this RCVMSG command is processed. If the message does not arrive in the specified time, the CL variables named to receive message fields are filled with blanks.

The program cannot wait for a message from a *program* message queue unless it is receiving a reply.

If a wait time is specified (not zero), the message queue is implicitly allocated to the first user whose message is received, and it is not released until the request has been handled by the program.

If a message is sent to a message queue in the same job, and the message queue is in break delivery mode, this parameter is ignored. (that implies WAIT(0), which is the default value for the WAIT parameter).

If the value specified for MSGKEY refers to an inquiry message, and MSGTYPE(*RPY) has been specified, the program ignores the WAIT parameter (Value for Wait is 0).

0: The program does not wait for the arrival of a message. If a message of the specified type is not in the queue when this command is processed, the specified CL variables are filled with blanks (or zeros, if they are decimal variables).

***MAX:** The program waits indefinitely for the arrival of the specified type of message.

number-of-seconds: Specifies the number of seconds that the program waits for the arrival of a specific type of message.

RMV

Specifies whether the message received by the program is removed from the message queue. For exception messages for unhandled exceptions, also specifies whether the exception is to be handled by the RCVMSG command. If MSGTYPE(*INQ) is specified, RMV(*NO) must also be specified so a reply to the inquiry message can be sent.

***YES:** The message is removed from the message queue. If the message is an unhandled exception, the exception is handled by running the RCVMSG command.

***NO:** The message is not removed from the message queue. It is left on the message queue as an old message. If the message is an unhandled exception, the exception is handled by running the RCVMSG command.

Note: Old messages are messages that have been received but not deleted. An old message can be received again in one of the following ways:

1. The message reference key of the message is specified for the MSGKEY parameter.
2. A message type of *FIRST, *LAST, *NEXT, or *PRV is specified for the MSGTYPE parameter.

***KEEPEXCP:** If the message is an exception message and the exception has not been handled, the exception is left unhandled and the message is left on the message queue as a new message. It can be received again by using the RCVMSG command to receive an *EXCP message. If the message is not an exception message, or if it is but the exception has already been handled, the message is left on the message queue as an old message.

To handle an exception after the RCVMSG has been run, the command can be run a second time by specifying RMV(*YES) or RMV(*NO).

Parameters for Received Message Fields

All of the following parameters are used to specify the names of the CL variables that receive the specified fields and attributes of a message when the message is received by the program. If WAIT (number-of-seconds) is specified, and the time-out occurs, the variables, which must already be declared in the program, are filled with blanks. If the message field returned is larger than the CL variable speci-

fied, the message field is truncated; if the message field is shorter, it is padded with blanks. If the program does not need the value for a specific message parameter, no CL variable is specified for it. If a parameter is not specified, the corresponding message value is not received in the program.

KEYVAR

Specifies the name of the CL character variable, if any, that contains the message reference key identifying the message received by the program containing this RCVMSG command. At the time the RCVMSG command is processed, the system returns the message reference key to the variable specified by KEYVAR in this command and changes the received message to an old message. The message reference key can then be used in the MSGKEY parameter in a subsequent RCVMSG command to receive the old message. If the message is not found, blanks are returned for the KEYVAR variable. For reply type messages, use the MSGKEY parameter on this command in conjunction with the KEYVAR parameter on the SNDPGMMSG command. The message reference key can also be used by this program for building message subfiles. The CL variable is the name of the field for which the SFLMSGKEY keyword is specified in the DDS for the message subfile.

Note: For message queues not associated with call stack entries, message reference keys can be used again after a message has been received and then removed (by specifying *YES on the RMV parameter).

The variable must be a character variable having a length of 4 characters.

Note: When using the message reference key (obtained from the CL variable specified by the KEYVAR parameter of the Send Program Message (SNDPGMMSG) command) to receive the reply to an inquiry message, note that the message reference key refers to the sender's copy. The sender's copy message is located on the reply message queue (which defaults to the program message queue that sent the inquiry message), not the message queue to which the inquiry message was sent.

MSG

Specifies the name of the CL character variable, if any, that contains the message text when the message is received by the program. This includes the message data fields that were substituted for substitution variables in the text before the message was sent (replies and impromptu messages contain no message data fields). This is a variable-length field, but most first-level message text is less than 132 characters in length. This parameter receives the reply from an *INQ message sent by the same program that issued the RCVMSG command.

MSGLEN

Specifies the name of the CL decimal variable, if any, that contains the total length of the first-level message text available to be received. The variable must be a decimal variable having a length of 5 positions.

SECLVL

Specifies the name of the CL character variable, if any, that contains the help text when the message is received by the program. This includes the message data fields that were substituted for substitution variables in the text before the message was sent (replies and impromptu messages do not have second-level messages). A message data field is a variable length field, but most help text is less than 3000 characters in length.

SECLVLLEN

Specifies the name of the CL decimal variable, if any, that contains the total length of the second-level message available to be received. The variable must be a decimal variable having a length of 5 positions.

MSGDTA

Specifies the name of the CL character variable, if any, that contains the message data record received by the program as part of the message. The message data record contains the substitution values (in a single character string) that are used in the text of the received message. The amount of data returned and its format depend on the message. Pointers contained in system messages are invalidated.

Note: If you use data that has an invalidated pointer in it an error message can occur.

MSGDTALEN

Specifies the name of the CL decimal variable, if any, that contains the total length of the message data record available to be received. The variable must be a decimal variable having a length of 5 positions.

MSGID

Specifies the name of the CL character variable, if any, that contains the message identifier of the message received by the program. If the message being received is an impromptu message, a message identifier is not returned. The minimum length of the variable is 7 characters.

SEV

Specifies the name of the CL decimal variable, if any, that contains the severity code of the message received by the program. The variable must be a decimal variable having a length of two positions. If the message being received is an impromptu message, the message severity is not returned.

SENDER

Specifies the name of the CL character variable, if any, that contains the identification of the sender of the message received through the RCVMSG command. The length of the CL variable depends on the SENDERFMT specification. If SENDERFMT(*SHORT)

is specified, the variable should be a minimum of 80 characters. If SENDERFMT(*LONG) is specified, the variable should be a minimum of 720 characters.

SENDERFMT

Specifies which format of the sender identification is returned. This parameter is valid only when the SENDER parameter is specified.

***SHORT:** The short format of the sender information is returned. The short format is 80 characters, with the last 9 characters set to blanks. The following information is returned:

- The first 26 characters identify the sending job
  - Job name (10)
  - User name (10)
  - Job number (6)
- The next 16 characters identify the sending program
  - Program name (12) (for an ILE procedure, this is the bound program name)
  - Instruction number (4) (for an ILE procedure, this field is set to blanks)
- The next 13 characters are the date and time
  - Date (7) (in the format 0yymmdd)
  - Time (6) (in the format hhmmss)
- The next 14 characters identify the sent-to call stack entry if the message is sent to a call message queue
  - Program name (10) (for an ILE procedure, this is the bound program name)
  - Instruction number (4) (for an ILE procedure, this field is set to blanks)
- The next 1 character identifies the sender type
  - "0" if sender is a program
  - "1" if sender is an ILE procedure
- The last 1 character identifies the sent-to type
  - "0" if receiver is a program
  - "1" if receiver is an ILE procedure

***LONG:** The long format of the sender information is returned. The long format is 720 characters, with the last 46 characters set to blanks. The following information is returned:

- The first 26 characters identify the sending job
  - Job name (10)
  - User name (10)
  - Job number (6)
- The next 13 characters are the date and time
  - Date (7) (in the format 0yymmdd)
  - Time (6) (in the format hhmmss)
- The next 1 character identifies the sender type
  - "0" if sender is a program
  - "1" if sender is an ILE procedure
- The next 1 character identifies the sent-to type
  - "0" if receiver is a program
  - "1" if receiver is an ILE procedure
- The next 12 characters are the sender's program name (for an ILE procedure, this is the bound program name)
- The next 10 characters are the sender's module name (if the sender is not an ILE procedure, this field is set to blanks)
- The next 256 characters are the sender's procedure name (if the sender is not an ILE procedure, this field is set to blanks)
- The next 1 character is blank
- The next 4 characters are the number of statement numbers available

  Note: A statement number represents a point in the sending program at which the message was sent. For programs and non-optimized procedures, this count is always 1. For optimized procedures, this count can be greater than 1, and each statement number represents a point at which the message could have been sent. If it is not possible to return statement numbers, this count will be 0.

- The next 30 characters return a maximum of 3 statement numbers, 10 characters each
- The next 300 characters return program or procedure information if the message being received was originally sent to a message queue associated with a call stack entry (otherwise, this field is set to blanks)
  - Sent-to program name (10) (for an ILE procedure, this is the bound program name)
  - Sent-to module name (10) (if the sender is not an ILE procedure, this field is set to blanks)
  - Sent-to procedure name (256) (if the sender is not an ILE procedure, this field is set to blanks)
  - Blanks (10)
  - Number of statements available for the receiving call stack entry (4)

    Note: A statement number represents a point at which the sent-to program was suspended (for example, due to a call operation) at the time the message was sent. For programs and non-optimized procedures, this count is always 1. For optimized procedures, this count can be greater than 1, and each statement number represents a point at which the message could have been sent. If it is not possible to return statement numbers, this count will be 0.

  - Statement numbers (30) (a maximum of 3 statement numbers, 10 characters each)

RTNTYPE

Specifies the name of the CL variable, if any, that contains the type code for the message received by the program. The variable must be a character variable having a length of 2 positions. The following values are returned to indicate the message type:

Value	Message Type
01	Completion
02	Diagnostic

04	Information
05	Inquiry
06	Sender's Copy
08	Request
10	Request with prompting
14	Notify (exception already handled at time of RCVMSG)
15	Escape (exception already handled at time of RCVMSG)
16	Notify (exception not handled at time of RCVMSG)
17	Escape (exception not handled at time of RCVMSG)
21	Reply, not checked for validity
22	Reply, checked for validity
23	Reply, message default used
24	Reply, system default used
25	Reply, from System Reply List

ALROPT

Specifies the name of the CL variable, if any, used to return the alert option of the message received by the program. The variable must be a character variable 9 positions in length.

MSGF

Specifies the name of the CL variable, if any, used to return the message file name of the message received by the program. If the message received is a stored message, the message file name of the file containing the stored message is returned. If the received message is not a stored message, the message file name is returned as blanks. The variable must be a character variable 10 positions in length.

Note: The message file name returned on this parameter is the message file specified or defaulted on either the SNDPGMMSG or SNDUSRMSG command, not the overriding message file. If an override was specified when sending the message, the same override should be used when receiving the message.

MSGFLIB

Specifies the name of the CL variable, if any, used to return the message file library name of the message received by the program. If the message received is a stored message, the message file library name specified when the message was sent is returned. If *LIBL was specified on the send command, *LIBL is returned. If the received message is not a stored message, the message file library name is returned as blanks. The variable must be a character variable 10 positions in length.

Note: The message file library name returned on this parameter is the message file specified or defaulted on either the SNDPGMMSG or SNDUSRMSG command, not the overriding message file library. If an override was specified when sending the message, the same override should be used when receiving the message.

SNDMSGFLIB

Specifies the name of the CL variable, if any, used to return the name of the library that contains the message file used to send the message. If the message received is stored in a message file, the name of the library containing the message file is returned. For example, if the name of the library containing the message file is MYLIB, and *LIBL is used to send a predefined message, this parameter would return MYLIB. If the message received is not a stored message, the message file library name is returned as blanks.

Examples

Example 1: Receiving a Message

```
RCVMSG  MSGQ(SMITH)  MSGKEY(&KEY)  MSG(&WORK)
```

This command receives the message having the message reference key specified by the program variable &KEY from the message queue SMITH. The text of the message is copied into the CL variable &WORK.

Example 2: Receiving a New Message

```
RCVMSG  MSGQ(INV)  WAIT(120)  MSG(&WORK)
```

This command receives a new message from the message queue named INV into the CL variable &WORK. The program waits no more than 120 seconds for the arrival of a new message if there are no new messages in the message queue. If there is more than one new message in the queue, the first message in the queue is the message received by the program.

Example 3: Receiving a Message From a Procedure

```
RCVMSG   PGMQ(*SAME CURRENT_MONTH_TOTALS)  MSGTYPE(*EXCP)
  RMV(*KEEPEXCP)  MSGID(&MID)  MSG(&MTEXT)
```

This command receives an exception message from the procedure CURRENT_MONTH_TOTALS. Since the specified name is more than 10 characters, the system does search for any programs. If the message is an unhandled exception, the message is left on the call message queue as a new message and the exception is not handled by the RCVMSG command. The message ID is returned in the CL variable &MID and the message text in the CL variable &MTEXT. To handle the exception and remove the message, run the following RCVMSG command:

```
RCVMSG   PGMQ(*SAME CURRENT_MONTH_TOTALS)
  MSGTYPE(*EXCP)  RMV(*YES)
```

Example 4: Receiving a Message from a Program or Procedure

```
RCVMSG   PGMQ(*SAME TARGETPGM)  MSGTYPE(*EXCP)
  RMV(*NO)  MSGID(&MID)  MSG(&MTEXT)
```

This command receives an exception message from the message queue of the program or procedure named TARGETPGM. Since the specified name is only 9 characters, the system searches both programs and procedures.

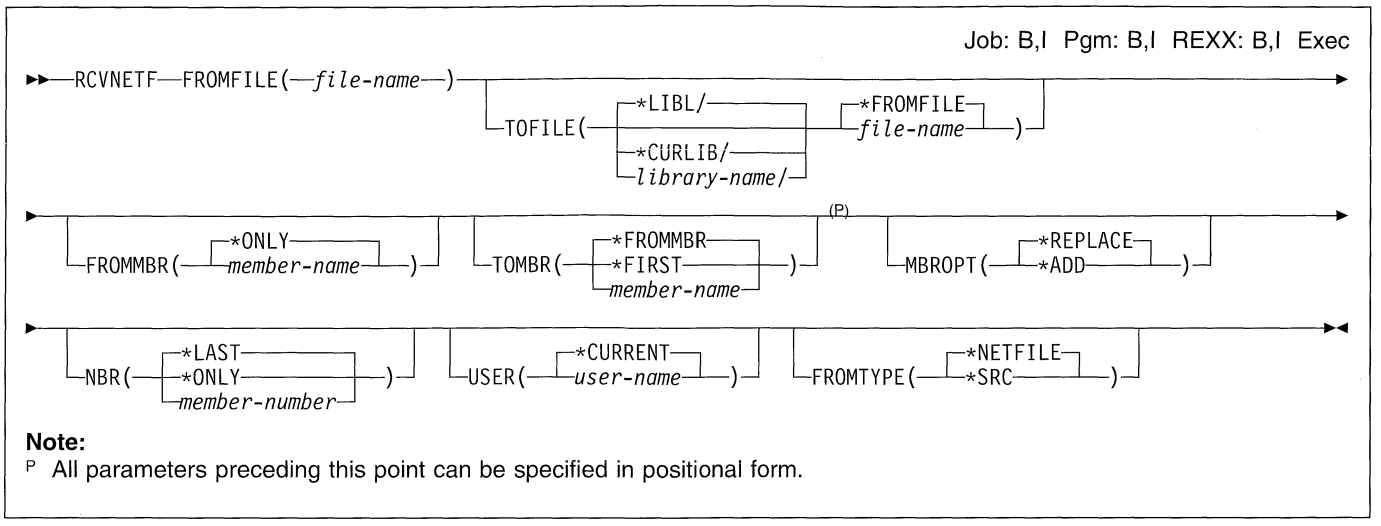
| Because RMV(*NO) is specified, if the message is an unhandled exception, the exception is handled by the RCVMSG command. The message is left on the message queue as an old message.

| Example 5: Receiving a Message Using Qualifiers

```
| RCVMSG   PGMQ(*SAME PRINT_RPT_FMT1 DEPTRPTS AREARPTS)
|   MSGTYPE(*EXCP)  RMV(*YES)
|   MSGID(&MID)  MSG(&MTEXT)
```

| This command receives an exception message from the message queue of the procedure named PRINT_RPT_FMT1. The procedure must have been compiled into the module DEPTRPTS and have been bound into the bound program AREARPTS. Since RMV(*YES) is specified, the exception is handled if the exception message is for an unhandled exception. The message is always removed from the message queue.

RCVNETF (Receive Network File) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Receive Network File (RCVNETF) command receives a network file and copies the records into a physical database file or a save file.

If the original file is a save file, it must be received into a save file. Once the file has been received, it is removed from the queue of network files. Before a file can be received, the file specified by the TOFILE parameter must already exist.

When a source physical file is sent, the source sequence number and change date in positions 1 through 12 of the record are sent with the file. These are kept if the file is received into a source physical file, and are truncated if the file is received into a nonsource physical file. When a file that was originally a nonsource physical file is received into a source physical file, the source sequence numbers are created and placed in front of the records.

If the file is a physical file, the record length of the to-file must be at least as large as the record length of the original file. If the record length of the to-file is larger than that of the original file, the records are padded to the end with the default record value for the to-file.

Restrictions:

1. A user with security officer authority can receive the files sent to any user. Users with other than security officer authority can receive only files sent to them or to their group profile.
2. The user must have read authority to the library containing the to-file, and use and add authority to the to-file. The following additional authority may be required:
   - Object management authority, if a member is added to the file.
   - Object management authority and delete authority, if a save file or existing physical file member is cleared.

Required Parameter

FROMFILE
Specifies the name of the file that is received. This is the name of the file on the sending system.

Optional Parameters

TOFILE
Specifies the qualified name of the file that receives the copied records.

Note: If the network file sent is a database file, the to-file must be a physical database file with a record length at least as large as that of the original file. If the file sent is a save file, the to-file must be a save file. Overrides to this file are ignored.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

***FROMFILE:** The network file is received into a file of the same name as the file sent.

file-name: Specify the name of the file into which the network file is received.

FROMMBR
Specifies the name of the file member that is received.

***ONLY:** Only one member is received for this file. This parameter value is valid only if there is one member for the specified network file, or if the NBR parameter is used to uniquely identify a single member.

member-name: Specify the name of the member that is received. A member name cannot be specified if the file is a save file.

TOMBR
Specifies the database file member that receives the data.

***FROMMBR:** The data is received into a member with the same name as the member specified in the FROMMBR parameter.

***FIRST:** The first member in the file receives the copied records.

member-name: Specify the name of the member that receives the records. A member name cannot be specified if the file is a save file.

MBROPT
Specifies whether the new records replace or are added to the existing records.

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

NBR
Specifies the number of the file member that is received. This number is used to identify the member that is received when there is more than one member of the same name available for the file.

***LAST:** The last network file member with the specified member name is received. The last member is determined as the last member to arrive at the user's system; both FROMFILE and FROMMBR parameter values are used to determine the last network file member.

Note: The file member that arrived last at the user's system may not have been the last one sent by the sending user. The network does not guarantee the arrival sequence of separately sent files.

***ONLY:** Only one file member of the specified file name is received. If more than one member of that name is available, an escape message is sent, and the command is not run.

member-number: Specify the number of the member that is received.

USER
Specifies the user to whom the file was sent.

***CURRENT:** The user profile under which the current job is running is used.

user-name: Specify the name of the user to whom the files were sent. Only a user with security officer authority can specify a name other than the user's own or the user's group profile.

FROMTYPE
Specifies the type of file that is received. This option should be used mainly when the file is an AS/400 system or System/38 source file that was sent by a System/370 Virtual Machine (VM) or Multiple Virtual Storage (MVS) user. Since VM or MVS cannot identify whether the file is a source file, the user must specify that the file is a source file or a nonsource file.

***NETFILE:** The network file type is used to determine whether file type conversion is needed.

If the file is a nonsource file and is:

- Received into a nonsource file, the file is received unchanged.
- Received into a source file, the sequence numbers and date fields are added.

If the file is a source file and is:

- Received into a nonsource file, the sequence numbers and date fields are removed (the first 12 bytes of each record).
- Received into a source file, the file is received unchanged.

***SRC:** The file being received is a source file. The sequence numbers and date fields are in the file. If the file is received into another source file, the sequence numbers and date fields are not added to the file being received. If the file is received into a nonsource file, the sequence numbers and date fields are removed from the file.

Note: *SRC must *not* be specified if the network file does not contain sequence numbers and date fields in the first 12 bytes of each record.

Examples

Example 1: Receiving a Member

```
RCVNETF   FROMFILE(FILEA)  TOFILE(FILEB/FILEA)
  FROMMBR(PAYROLL)
```

This command receives member PAYROLL of file FILEA into member PAYROLL of file FILEA in library FILEB. If there is an existing member of that name, the records in the member are replaced. If multiple members of that name are available, the last one to arrive at the destination system is received.

Example 2: Receiving a Network File

```
RCVNETF   FROMFILE(PERSONNEL)  NBR(*LAST)  USER(USR1)
```

This command receives a network file named PERSONNEL, which was sent to user USR1, into a file with the same

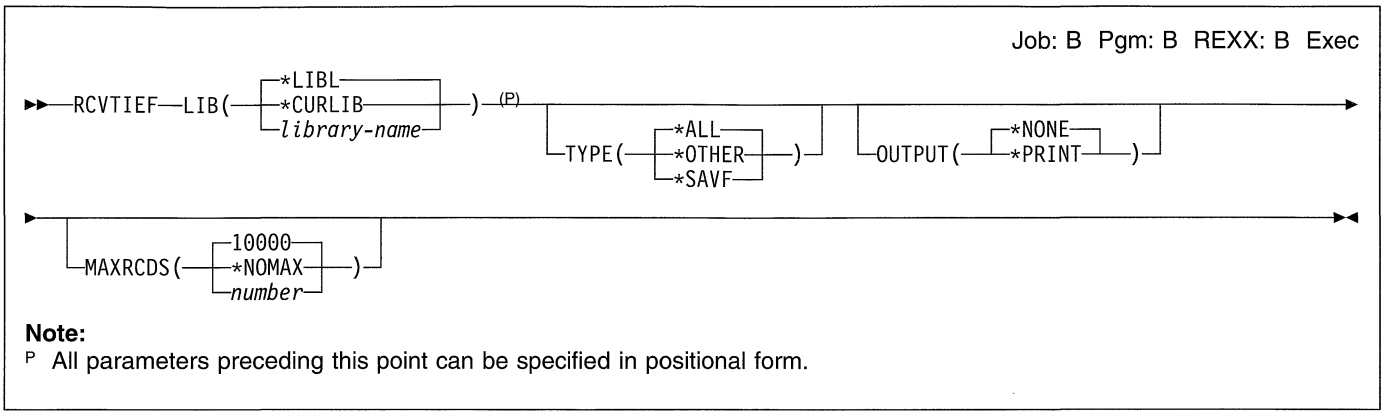
name. Because the FROMMBR parameter is not specified, there must be only one member name available for this file. Because USR1 is specified, only someone with a user profile of USR1, someone with a group profile of USR1, or someone with security officer authority can use this command.

Example 3: Receiving a Source File

```
RCVNETF   FROMFILE(FILEA)  TOFILE(FILEB/FILEA)
  FROMMBR(PAYROLL)  FROMTYPE(*SRC)
```

This command specifies that the file being received is a source file and the sequence numbers and date fields are not added to the file being received.

RCVTIEF (Receive Technical Information Exchange File) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Receive Technical Information Exchange File (RCVTIEF) command receives files transmitted from the remote support network.

Required Parameters

LIB

Specifies the library where the files are stored.

***LIBL:** The library list is used to locate the file.

***CURLIB:** The current library is used to locate the file. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library where the file is located.

Optional Parameters

TYPE

Specifies the types of files that are received.

***ALL:** All available files are received.

***OTHER:** Files with unspecified contents are received.

***SAVF:** Save files are received.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***NONE:** A list of files received is not printed.

***PRINT:** The output is printed with the job's spooled output.

MAXRCDS

Specifies the maximum size (number of records) of any file that can be received.

10000: The maximum file size is 10000 records.

***NOMAX:** The system maximum is used.

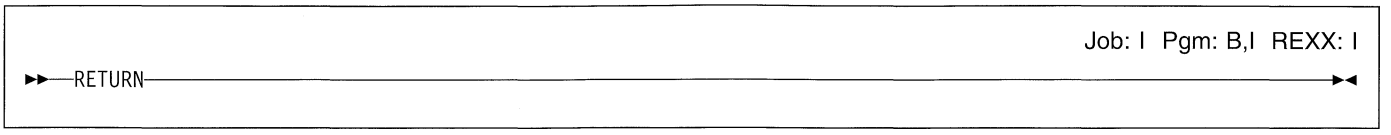
number: Specify the maximum file size that can be received.

Example

```
RCVTIEF  LIB(MAIL)  TYPE(*OPEN)  OUTPUT(*PRINT)
  MAXRCDS(1000)
```

This command receives from TIE all OPEN files (any file except a save file). A list of the received files is printed. If any of the received files are larger than 1000 records, the RCVTIEF command fails. If all OPEN files are received successfully, they are removed from the mailbox.

RETURN (Return) Command



Purpose

The Return (RETURN) command returns control either to the next higher call stack entry in the call stack or to the subsystem monitor that controls the job.

When used outside a CL program, this command performs the same function as the CF1 key. It returns control from the most recent invocation of QCMD (the IBM-supplied control language processor that interprets and processes CL commands for the system) back to the outside program manager. When used in a CL program, this command returns control to the next command or high-level language statement in the calling program at the point where it called the returning program. If this command is used in the highest invocation level in the routing step (either the QCMD program, which is the interpretive CL command processor, or a CL program), the routing step is ended.

Note: If the RETURN command is entered interactively from the highest recursion level while the subsystem is undergoing a controlled end resulting from

- An End Subsystem (ENDSBS) command
- An End System (ENDSYS) command
- A Power Down System (PWRDWNSYS) command)

end-of-job processing occurs unless you receive the inquiry message and indicate that you want to return to the command entry display.

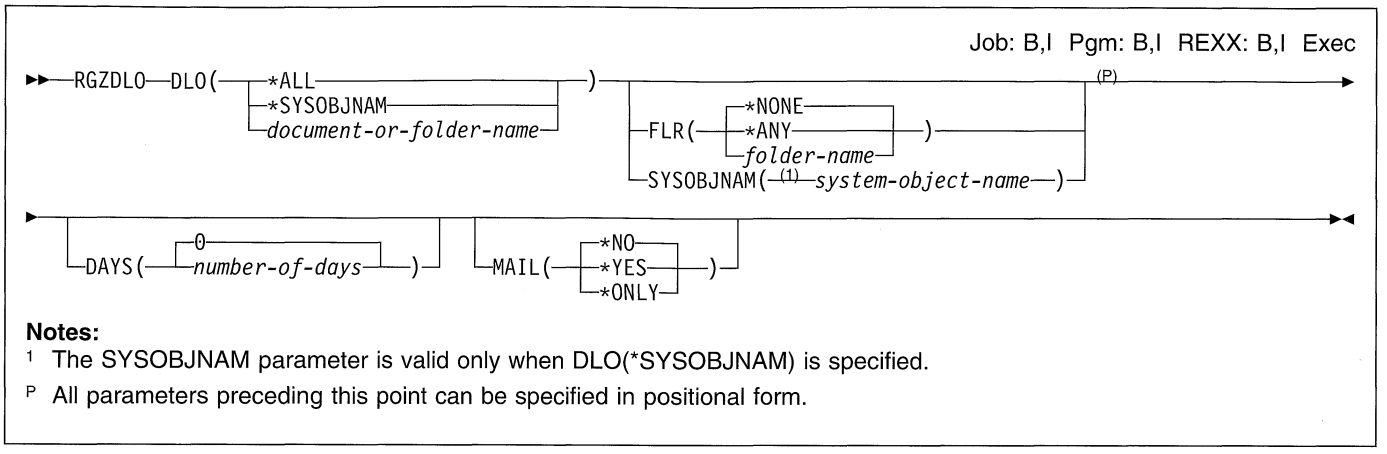
There are no parameters for this command.

Example

```
RETURN
```

When used in a CL program, this command returns control to the CL command or high-level language statement immediately following the point in the last calling program at which this program was called. When used in a an interactive job, this command returns control to the next higher level of QCMD. If the RETURN command is run in the highest call level program (QCMD) in the routing step, an inquiry message is sent, and the user has the option of returning to the command entry display. Otherwise, the routing step ends as usual.

RGZDLO (Reorganize Document Library Object) Command



Notes:
1 The SYSOBJNAM parameter is valid only when DLO(*SYSOBJNAM) is specified.
P All parameters preceding this point can be specified in positional form.

Purpose

The Reorganize Document Library Object (RGZDLO) command allows the user to reorganize five types of objects:

1. All folderless documents.
2. All unfiled mail documents.
3. All filed documents and folders.
4. All documents and folders within a specified folder.
5. Individual documents or folders specified by a folder name, document name, or system-object-name.

When a document is reorganized, all unused storage is removed Some major causes of unused storage is adding, deleting, or editing information.

Restrictions:

1. The user must have *ALLOBJ or *SECADM special authorities to specify DLO(*MAIL) or to specify DLO(*ALL) and FLR(*ANY) together.
2. To reorganize a document or folder, the user must have *ALLOBJ or *SECADM special authority or at least *CHANGE authority to the document or folder and be enrolled in the system directory.
3. To reorganize a document or folder, the user must have exclusive use of the document or folder.
4. No other jobs that use documents or folders can be running while mail documents that have not been filed are being reorganized.

Required Parameters

DLO

Specifies the document library object being reorganized.

***ALL:** All document library objects are reorganized. If FLR(*NONE) is specified with this parameter, all folderless documents are reorganized. If FLR(*ANY) is specified with this parameter, all filed documents and folders are reorganized. If MAIL(*YES) is specified with this parameter, all unfiled mail documents as well as all filed documents and folders are reorganized. If MAIL(*ONLY) is specified with this parameter, only unfiled mail documents are reorganized. If FLR(*folder-name*) is specified with this parameter, all folders and documents within it are reorganized.

***SYSOBJNAM:** A system object name specified on the SYSOBJNAM parameter is used to identify the document or folder being reorganized.

document-or-folder-name: Specify the name of the document or folder being reorganized. The FLR parameter also can be used to reorganize a document by specifying reorganization of:

- The folder that contains the document being reorganized
- The folder that contains the nested folder that contains the document being reorganized

Optional Parameters

FLR

Specifies the name of the folder that contains the document.

Note: If the document does not exist in a folder, *NONE is specified.

***NONE:** The document or folder is not located in a folder. If DLO(*ALL) is specified, this refers to all folderless documents. If DLO(*document-or-folder-name*) is specified, this refers to a first-level folder.

***ANY:** The documents and folders being reorganized are not qualified by any folder. This value is valid only when DLO(*ALL) is specified.

folder-name: Specify the folder name that contains the documents or folders.

SYSOBJNAM

Specifies the system object name. This parameter is valid only when DLO(*SYSOBJNAM) or DOCL(*SYSOBJNAM) is specified. A full ten characters must be specified.

DAYS

Specifies the number of days that must elapse since a document was last referred to before it is eligible for reorganization.

0: Any document may be reorganized.

number-of-days: Specify the number of days that elapsed since a document or folder was last used before it is eligible for reorganization.

MAIL

Specifies whether objects being reorganized include, omit, or are limited to mail documents that have not been filed.

***NO:** Mail documents that have not been filed are not reorganized.

***YES:** Mail documents that have not been filed are reorganized along with other folders and documents. This value is valid only when DLO(*ALL) and FLR(*ANY) are specified.

***ONLY:** Only mail documents that have not been filed are reorganized. This value is valid only when DLO(*ALL) and FLR(*ANY) are specified.

Examples

Example 1: Reorganizing Folders and Documents

```
RGZDLO   DLO(*ALL)  FLR(*ANY)
```

This command reorganizes all filed folders and documents that exist on the system.

Example 2: Reorganizing Folders, Documents, and Unfiled Mail

```
RGZDLO  DLO(*ALL)  FLR(*ANY)  MAIL(*YES)
```

This command reorganizes all filed folders, documents, and all unfiled mail documents that exist on the system.

Example 3: Reorganizing Unfiled Mail Documents

```
RGZDLO  DLO(*ALL)  FLR(*ANY)  MAIL(*ONLY)
```

This command reorganizes all unfiled mail documents that exist on the system.

Example 4: Reorganizing Folderless Documents

```
RGZDLO  DLO(*ALL)  FLR(*NONE)
```

This command reorganizes all folderless documents that exist on the system.

Example 5: Reorganizing Documents Within Folders Within Folders

```
RGZDLO  DLO(*ALL)  FLR(FLRA)
```

This command reorganizes all documents within folders contained in folder FLRA, then the folders within folder FLRA are reorganized.

Example 6: Reorganizing an Individual Document or Folder

```
RGZDLO  DLO(*SYSOBJNAM)  SYSOBJNAM(DCN1371951)
```

This command reorganizes the individual document or folder identified by the SYSOBJNAM object.

Example 7: Reorganizing a Document

```
RGZDLO  DLO(DOC1)  FLR(FLRA)
```

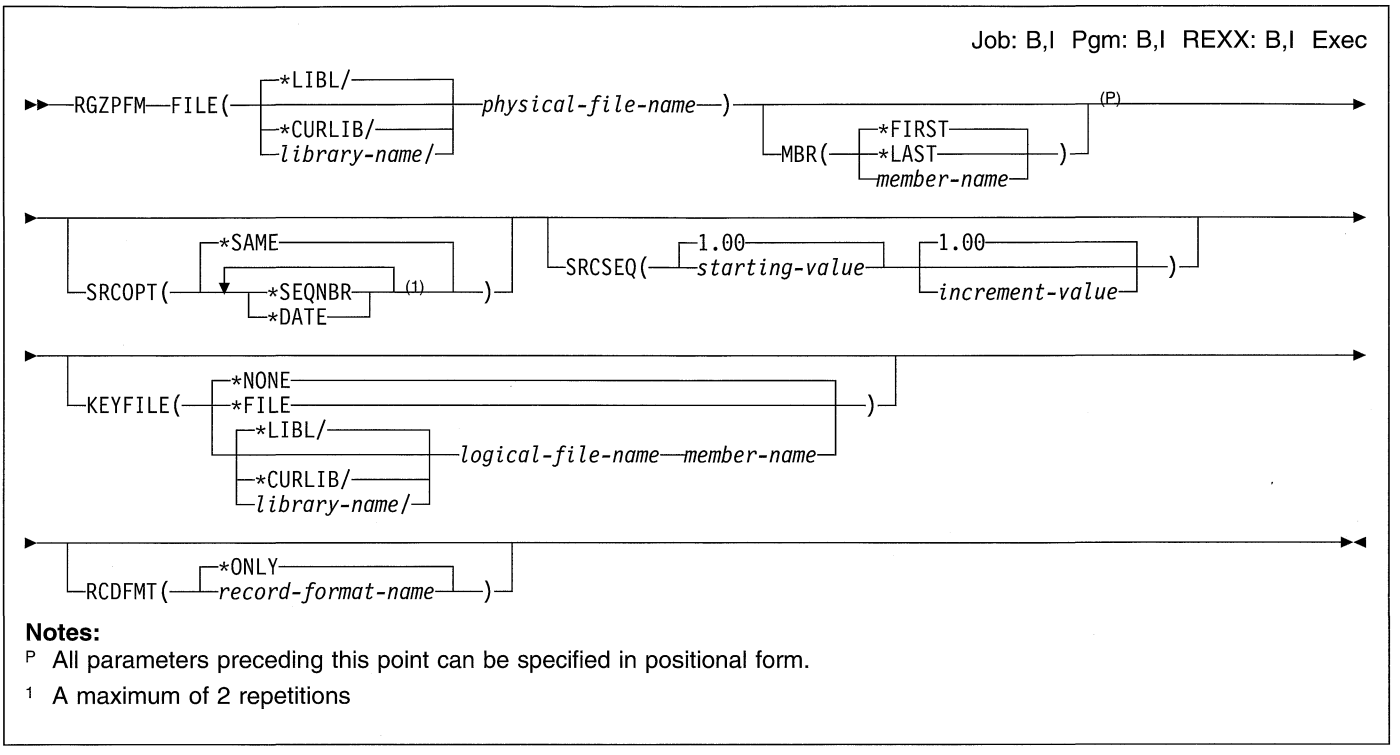
This command reorganizes the document named DOC1 in folder FLRA.

Example 8: Reorganizing Documents Not Referenced

```
RGZDLO  DLO(*ALL)  FLR(*ANY)  DAYS(30)
```

This command reorganizes all filed documents and folders that have not been referenced in the past 30 days.

RGZPFM (Reorganize Physical File Member) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 A maximum of 2 repetitions

Purpose

The Reorganize Physical File Member (RGZPFM) command compresses (removes deleted records) one member of a physical file in the database, and it optionally reorganizes that member.

If a keyed file is identified in the KEYFILE parameter, the system reorganizes the member by changing the physical sequence of the records in storage to either match the keyed sequence of the physical file member's access path, or to match the access path of a logical file member that is defined over the physical file. Reorganization can decrease file processing time when a program is reading sequentially through a keyed physical file or through a keyed logical file.

When the member is reorganized and the KEYFILE parameter is specified, the sequence in which the records are actually stored is changed, and any deleted records are removed from the file. If the KEYFILE parameter is not specified, the sequence of the records does not change, but deleted records are removed from the member. Optionally, new sequence numbers and zero date fields are placed in the source fields of the records. These fields are changed after the member has been compressed or reorganized.

Notes:

1. If you cancel this command, you must rebuild all the access paths for the member.
2. The RGZPFM command ignores all file overrides that are currently in effect for the job. The file names specified in the FILE and KEYFILE parameters identify the files actually used in the reorganize operation, regardless of overrides that may exist for these files.

Restriction: During the reorganization of a physical member, the file being reorganized is locked (similar to an *EXCL lock with no time-out) for the time of the RGZPFM command so that no access is possible. Concurrent attempts by another job to use a function that refers to the file may result in a "lock up" for that work station until the running of the RGZPFM command is completed. Examples of commands that cannot be used on a file that is concurrently being reorganized are:

- WRKACTJOB (Work with Active Jobs) (Select 11-Locks; Select 1-WRKOBJLCK)
- DSPDBR (Display Database Relations)
- DSPFD (Display File Description)
- DSPFFD (Display File Field Description)
- DSPJOB (Display Job) (Option 12-Display Locks; Option 14-Display Open files; F10-Job record locks)
- WRKJOB (Work with Job) (Option 12-Display Locks; Option 14-Display Open files; F10-Job record locks)
- WRKLIB (Work with Library) (The library that contains the file being reorganized)
- DSPOBJD (Display Object Description)

- WRKOBJLCK (Work with Object Locks)
- DSPRCDLCK (Display Record Locks)
- WRKOBJ (Work with Object Descriptions)
- DSPLIB (Display Library) (The library that contains the file being reorganized)
- WRKF (Work with Files)
- Any other function that refers to the file being reorganized

Required Parameters

FILE

Specifies the qualified name of the physical file whose member is being reorganized.

The name of the physical file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

physical-file-name: Specify the name of the physical file.

Optional Parameters

MBR

Specifies the name of the member in the file being reorganized.

***FIRST:** The first member in the database file is used.

***LAST:** The last member of the specified physical file is reorganized.

member-name: Specify the name of the file member being reorganized.

SRCOPT

Specifies, for physical *source* files only, whether the member places new numbers in the sequence number field, places zeros in the date field, or changes both fields. Changes occur after the records are compressed or reorganized.

***SAME:** The value does not change.

***SEQNBR:** The records have a new sequence number placed into the sequence number field. The SRCSEQ parameter specifies a start value and an increment value. If *SEQNBR is specified, *DATE can also be specified.

***DATE:** The records have a null date (000000) placed in the date field. If *DATE is specified, *SEQNBR can also be specified.

SRCSEQ

Specifies, only when SRCOPT(*SEQNBR) is also specified, the sequence number that is given to the first record in the source file member and the increment value that is used to renumber all other records in the member. If the member is renumbered but SRCSEQ is not specified, SRCSEQ(1.00 1.00) is assumed; the source records are renumbered sequentially starting with 1.00, and the whole number increment of 1 is used.

Element 1: Starting Value

1.00: The first source record in the member has a sequence number of 0001.00.

starting-value: Specify the sequence number (ranging from 0000.01 through 9999.99) of the first source record in the member. A whole number of no more than four digits and/or a fraction of no more than two digits can be specified. If the starting value contains a fraction, a decimal point must be used. Examples are .01 and 3250.4. If a value has a fraction of .00, such as 5000.00, it can be coded without the fraction; either 5000 or 5000.00 is valid.

Element 2: Increment Value

1.00: The source records are renumbered in the member with whole number increments of 1 (for example, 1.00, 2.00, 3.00 and so on).

increment-value: Specify the increment value (ranging from 0000.01 through 9999.99) for renumbering all source records following the first record. A whole number of no more than four digits and/or a fraction of no more than two digits can be specified. If the increment value contains a fraction, a decimal point must be used. For example, if SRCSEQ(5000 10) is specified, the first record in the reorganized member is numbered 5000.00, the second is 5010.00, the third is 5020.00, and so on. If SRCSEQ(*N .25) is specified, the records are numbered 1.00, 1.25, 1.50, 1.75, 2.00, and so on. If a starting value of .01 and an increment value of .01 are specified, there are 999,999 unique sequence numbers possible. If the maximum sequence number of 9999.99 is reached, the remaining records are also assigned the sequence number 9999.99.

KEYFILE

Specifies whether the physical file member has its arrival sequence changed to match its keyed sequence, is reorganized in the sequence of a logical file member, or is not reorganized. If KEYFILE specifies a multiple-format logical file and member, the RCDFMT parameter must also be specified.

Note: Join logical files cannot be specified as key files, and a logical file in this parameter is not allowed to have a select/omit access path.

***NONE:** The member is not reorganized; it is only compressed by having its deleted records removed.

***FILE:** For a physical file member having a keyed sequence access path, the arrival sequence of the records in the member is changed to match their keyed sequence.

Element 1: Logical File Name

The name of the logical file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

logical-file-name: Specify the qualified name of the logical file whose member's sequence is used to reorganize the physical file member.

Element 2: Member Name

member-name: Specify the name of the member whose sequence is used to reorganize the physical file member.

RCDFMT

Specifies the record format name if the physical file member is reorganized in the sequence of a multiple-format logical file.

***ONLY:** The logical file specified by the KEYFILE parameter has only one record format, which is used to reorganize the physical file member.

record-format-name: Specify the name of a record format in the multiple-format logical file that is used to reorganize the physical file member.

Note: Compression of a file occurs when the space occupied by a deleted record is freed to hold a record that is not deleted.

Examples

Example 1: Reorganizing by Deleting Records

```
RGZPFM   FILE(PAYROLL)  MBR(MBR1)
```

This command compresses member MBR1 of the PAYROLL file by removing the deleted records from the file member.

Example 2: Reorganizing in Keyed Sequence

```
RGZPFM   FILE(QCLSRC)  MBR(CLMBR2)
  SRCOPT(*SEQNBR *DATE)  KEYFILE(*FILE)
  SRCSEQ(1.00 .25)
```

This command reorganizes the member CLMBR2 of the CL source file QCLSRC in keyed sequence, with the sequence number field used as the key. The reorganized member has new sequence numbers (starting at 1.00 and incrementing by .25) and a null date (000000) placed in all records when the original member is reorganized.

RLSCMNDEV (Release Communications Device) Command

Job: B,I Pgm: B,I REXX: B,I Exec

Note:

(P) All parameters preceding this point can be specified in positional form.

Purpose

The Release Communications Device (RLSCMNDEV) command restores the communications capability of a specified device held by the Hold Communications Device (HLDCMNDEV) command.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Required Parameters

DEV

Specifies the name of the device whose communications are released after being held. Specify the name of the device. Devices whose communications can be held by the HLDCMNDEV command are:

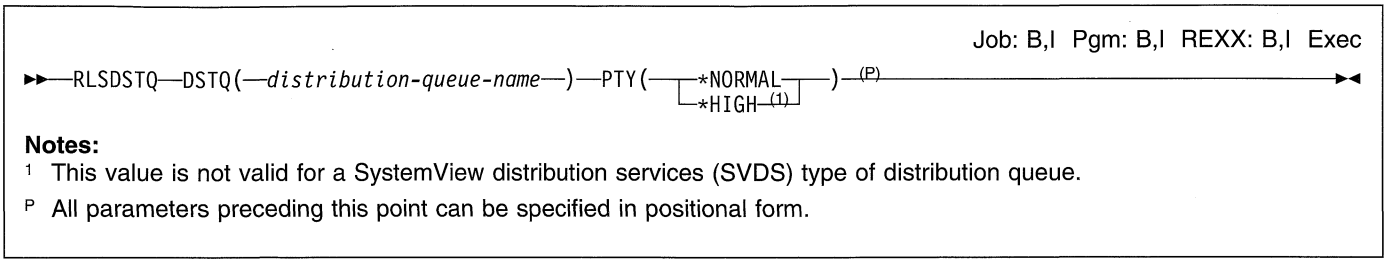
DEV Value	Device
3180	Display station
3277	Display station
3278	Display station
3279	Display station
3287	Printer (work station)
5219	Printer (work station)
5224	Printer (work station)
5225	Printer (work station)
5251	Display station
5252	Display station
5256	Printer (work station)
5291	Display station
5292	Display station
PLU1	Primary logical unit, type 1 (for SNA)
BSC	Binary synchronous device (Base and RJE)
BSCT	This AS/400 system is a BSC multipoint tributary station
APPC	Logical unit in advanced program-to-program communications network

Example

```
RLSCMNDEV  DEV(WSPR05)
```

This command restores the communications capability of the currently held device WSPR05.

RLSDSTQ (Release Distribution Queue) Command



Notes:

1 This value is not valid for a SystemView distribution services (SVDS) type of distribution queue.

P All parameters preceding this point can be specified in positional form.

Purpose

The Release Distribution Queue (RLSDSTQ) command releases a distribution queue from a held status and allows it to be sent.

Distribution queue names are translated to the graphic character set and code page 930 500, using the job's coded character set identifier (CCSID).

Restrictions:

1. This command is shipped with public *EXCLUDE authority and the QPGMR and QSYSOPR user profiles have private authorities to use the command.
2. Messages that report errors about distribution queues may display or print different characters than the user entered for the distribution queue name because of internal system transformations. Similarly (depending on the language used for the work station), the internal value for a distribution queue name may differ from the characters shown on the Work with Distribution Queue (WRKDSTQ) command. An error may be reported if the character-string value specified for the DSTQ parameter does not match the rules for an internal distribution queue value or if it does not match the internal value for any defined distribution queue (ignoring case differences).

Required Parameters

DSTQ

Specifies the name of the distribution queue being released from being held. Both normal and high priority portions of the specified distribution queue are shown or printed. The queue specified must have been previously configured. See the Configure Distribution Services (CFGDSTSRV) command or the Add Distribution Queue (ADDDSTQ) command.

PTY

Specifies whether the normal priority or high priority portion of the specified queue is released from being held.

***NORMAL:** Releases the normal priority queue, which is for distributions with a service level of data low.

***HIGH:** Releases the high priority queue, which is for distributions with a service level of fast, status, or data high.

Examples

Example 1: Releasing the Normal Priority Portion of the Queue

```
RLSDSTQ  DSTQ(CHICAGO)  PTY(*NORMAL)
```

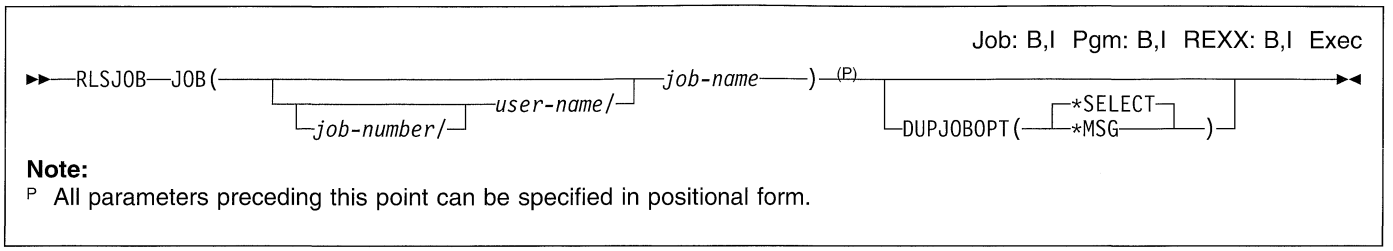
This command releases the normal priority portion of the CHICAGO distribution queue.

Example 2: Releasing the High Priority Portion of the Queue

```
RLSDSTQ  DSTQ(ATLANTA)  PTY(*HIGH)
```

This command releases the high priority portion of the ATLANTA distribution queue.

RLSJOB (Release Job) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Release Job (RLSJOB) command makes a job eligible for processing after that job has been held from processing by the HLDJOB (Hold Job) command or if the job was submitted to the system as a held job by the JOB or SBMJOB (Submit Job) commands. The job being released could have been on the job queue, output queue, or active in a subsystem (competing for system resources) when it was held. Spooled files that are held because SPLFILE(*YES) is specified in the HLDJOB command are also released.

Restriction: The job being released must belong to the user issuing the command or the user must have the special job control authority (*JOBCTL).

Required Parameters

JOB

Specifies the qualified name of the job being released. If no job qualifier is given, all of the jobs currently in the system are searched for the job name. If more than one of the specified names are found, a qualified job name must be specified.

A job identifier is a qualified name with up to three elements. For example:

```
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

job-name: Specify the name of the job being released.

user-name: Specify the name of the user of the job being released.

job-number: Specify the number of the job being released.

DUPJOBOPT

Specifies the action taken when duplicate jobs are found by this command.

***SELECT:** The selection display is shown when duplicate jobs are found during an interactive session. Otherwise, a message is issued.

***MSG:** A message is issued when duplicate jobs are found.

Examples

Example 1: Releasing a Job for Processing

```
RLSJOB    JOB(123456)
```

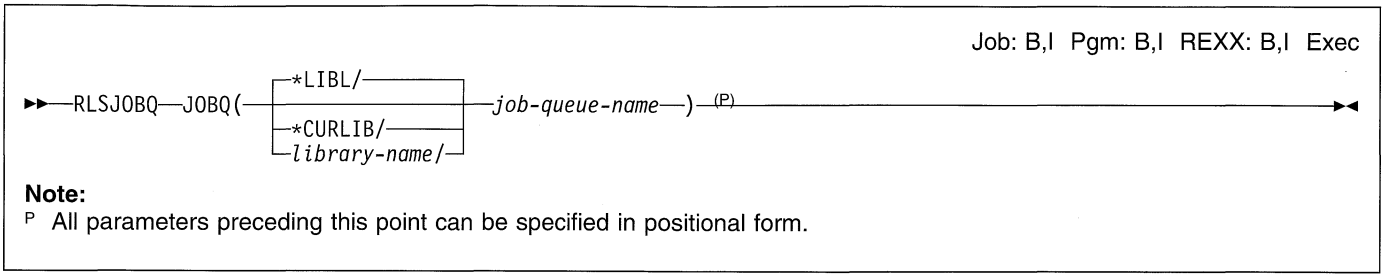
This command releases the job 123456 for processing. If the corresponding HLDJOB command had specified SPLFILE(*YES), any spooled files for job 123456 are also released.

Example 2: Releasing a Job for Processing

```
RLSJOB    JOB(DEPTXYZ/987654)
```

This command releases job name 987654 that was submitted by a user through the user profile DEPTXYZ and later held. The qualified form of the job name is used when jobs with duplicate names exist in the system.

RLSJOBQ (Release Job Queue) Command



Note:

| (P) All parameters preceding this point can be specified in positional form.

Purpose

The Release Job Queue (RLSJOBQ) command releases, for additional processing, the jobs on the specified job queue that were previously held by a HLDJOBQ (Hold Job Queue) command. If the jobs were held by something other than a HLDJOBQ command, they are not released.

Required Parameters

JOBQ

Specifies the qualified name of the job queue to be released for further processing.

| The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

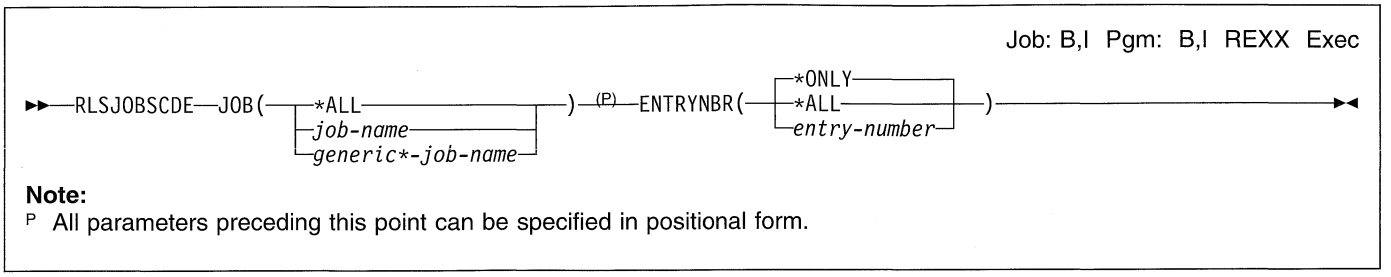
job-queue-name: Specify the name of the job queue.

Example

```
RLSJOBQ  JOBQ(QBATCH)
```

Jobs on the job queue QBATCH that were held by a HLDJOBQ command become eligible for processing, including jobs that were placed on the queue while it was being held. Specific jobs that were held by the HLDJOB command or that were put on the job queue in the held state are not released.

RLSJOBSCDE (Release Job Schedule Entry) Command



Note:
| P All parameters preceding this point can be specified in positional form.

Purpose

The Release Job Schedule Entry (RLSJOBSCDE) command allows you to release an entry, entries, or generic entries in the job schedule. Each job schedule entry contains the information needed to automatically submit a batch job one time, or at regularly scheduled intervals. If you release a job schedule entry, a job is not submitted immediately, even if the date and time at which it was scheduled to be submitted passed while the entry was held. The job is submitted on any future dates for which the entry is scheduled to be submitted.

Restriction: To release entries, you must have *JOBCTL special authority; otherwise you can release only those entries that you added.

Required Parameters

JOB

Specifies the name of the job schedule entry.

***ALL:** All of the job schedule entries for which you have authority are released. If JOB(*ALL) is specified, ENTRYNBR(*ALL) must also be specified.

job-name: Specify the name of the job schedule entry.

| *generic*-job-name:* Specify a generic name. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be released only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names." If a generic name is specified, ENTRYNBR(*ALL) must also be specified.

ENTRYNBR

Specifies the number of the job schedule entry you want to release. The message sent when an entry is successfully added contains the entry number. You can also determine the entry number by using the Work with Job Schedule Entries (WRKJOBSCDE) command. Press F11 from the Work with Job Schedule Entries display to show the entry numbers of the selected entries.

***ONLY:** One entry in the job schedule has the job name specified on the JOB parameter. If *ONLY is specified and more than one entry has the specified job name, no entries are released and a message is sent.

***ALL:** All entries with the specified job name are released.

entry-number: Specify the number of the job schedule entry you want to release.

Examples

Example 1: Releasing All Job Schedule Entries

```
RLSJOBSCDE  JOB(*ALL)  ENTRYNBR(*ALL)
```

This command releases all the job schedule entries.

Example 2: Releasing an Individual Job Schedule Entry

```
RLSJOBSCDE  JOB(PAYROLL)  ENTRYNBR(*ONLY)
```

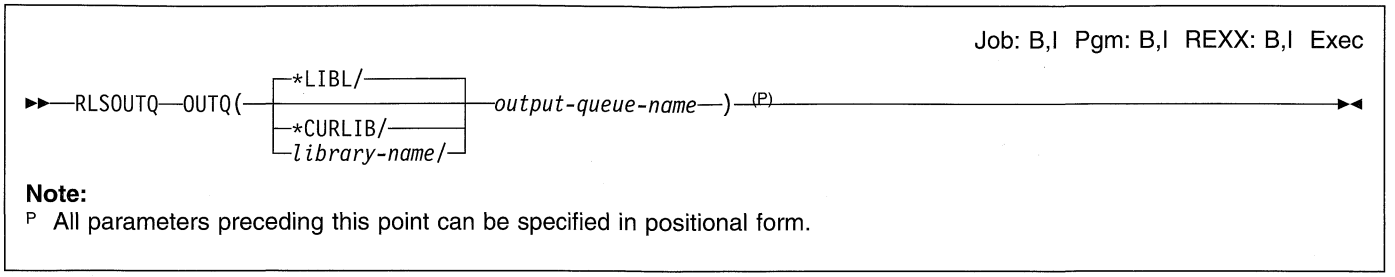
This command releases entry PAYROLL in the job schedule.

Example 3: Releasing a Generic Job Schedule Entry

```
RLSJOBSCDE  JOB(PAY*)  ENTRYNBR(*ALL)
```

This command releases all entries in the job schedule with the prefix PAY in their names.

RLSOUTQ (Release Output Queue) Command



Note:

| (P) All parameters preceding this point can be specified in positional form.

Purpose

The Release Output Queue (RLSOUTQ) command releases an output queue that was previously held by a HLDOUTQ (Hold Output Queue) command. This command allows all currently waiting spooled files, and all spooled files that are added to the output queue after the command is sent, to be processed by a spooling writer. Files held by a HLDSPLF (Hold Spooled File) command or created in a held state, are not released.

Required Parameters

OUTQ

Specifies the qualified name of the output queue.

| The name of the output queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

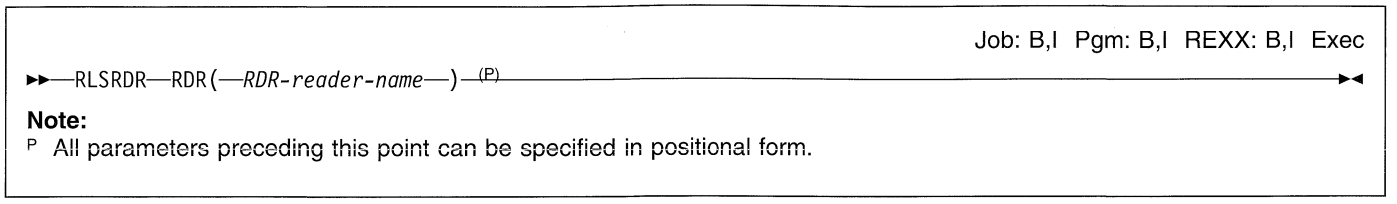
output-queue-name: Specify the name of the output queue being released.

Example

```
RLSOUTQ  OUTQ(PRINTER)
```

On the output queue named PRINTER, spooled files that were held by a HLDOUTQ command are released for further processing. This includes spooled files placed on the queue while it was being held, except for specific files that have been held by the HLDSPLF command or were put on the queue in hold.

RLSRDR (Release Reader) Command



Purpose

The Release Reader (RLSRDR) command releases the specified spooling reader making it available to again process jobs on the job queue. The specified reader was held by a previous HLDRDR (Hold Reader) command. Data is not lost.

Required Parameters

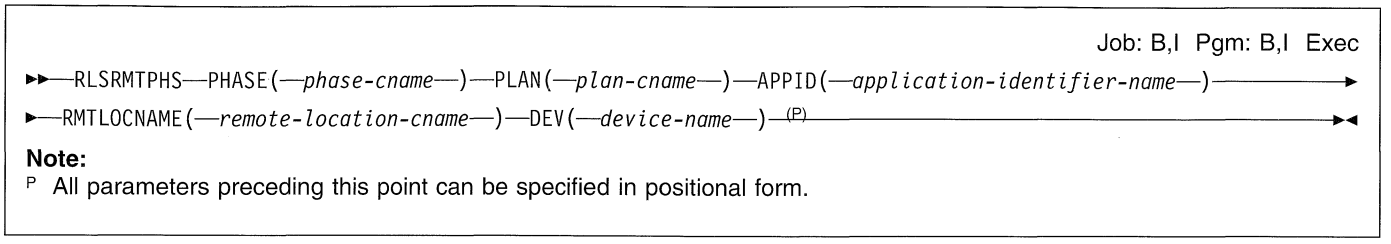
RDR
Specifies the name of the spooling reader being released. Specify the name of the spooling reader.

Example

```
RLSRDR  RDR(DISKETTE)
```

This command releases the diskette reader named DISKETTE for additional processing.

RLSRMTPHS (Release Remote Phase) Command



Purpose

The Release Remote Phase (RLSRMTPHS) command initiates a session between the AS/400 system and a System/370* NetView* Distribution Manager (NDM) host system. After the NDM releases the phase, or there is an unsuccessful attempt to release the phase, the session is ended.

The following considerations apply when running this command:

- The NDM phase specified by the PHASE parameter must exist and be part of the NDM plan specified by the PLAN parameter.
- The NDM phase specified by the PHASE parameter must be in a HELD state on the host system.
- The NDM plan specified by the PLAN parameter must exist and have been previously submitted to the NDM host application specified by the APPID parameter.
- The device specified by the DEV parameter must be a Systems Network Architecture upline facility (SNUF) device and must be program start request (PSR) capable.
- This command runs only on a node which is currently functioning as a host interface node to the NDM host system. However, it is not restricted to releasing only those NDM phases whose destination is the node sending the command. Any phase may be released for any node that shares the same host interface node sending this command.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR and QSYSOPR user profiles have private authorities to use the command.

Required Parameters

PHASE
Specifies the name of the NetView Distribution Manager phase that is released. This phase must exist on the NDM host system as part of the plan specified by the PLAN parameter, and must be in a HELD state.

PLAN
Specifies the name of the NetView Distribution Manager plan that contains the phase that is released. This plan must exist on the NDM host.

APPID
Specifies the name of the NetView Distribution Manager application under which the plan name specified by the PLAN parameter was submitted. This is the same name by which NDM was made known to MVS when it was generated.

RMTLOCNAME
Specifies the remote location name of the system with which this object communicates.

Note: The device with which the user's program is communicating is specified on the DEV parameter.

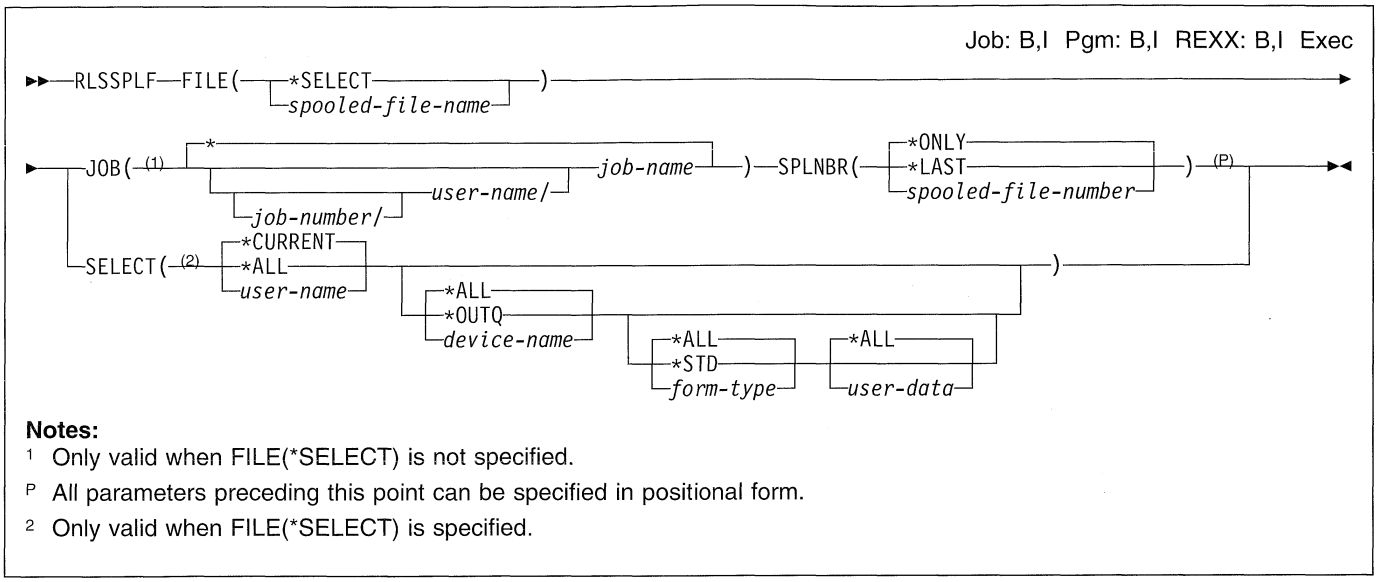
DEV
Specifies the device name of the AS/400 device that is used for the communications session started as a result of this command. The device must be a SNUF device and must be PSR capable.

Example

```
RLSRMTPHS  PHASE(MESSAGE)  PLAN(ALEXPLAN)  APPID(DSXNDM)
  RMTLOCNAME(A083187)  DEV(SNUFDEV)
```

This command initiates a session using device SNUFDEV with remote location name A083187 to connect with the System/370 NetView Distribution Manager host application DSXNDM. After the session connection is made, phase MESSAGE, as part of plan ALEXPLAN, attempts to release. If the release is successful, message CPC8889 (Phase MESSAGE released by NetView Distribution Manager) is sent. If the release is not successful, message CPF8880 (Phase MESSAGE not released by Netview Distribution Manager) is sent.

RLSSPLF (Release Spooled File) Command



Notes:
1 Only valid when FILE(*SELECT) is not specified.
P All parameters preceding this point can be specified in positional form.
2 Only valid when FILE(*SELECT) is specified.

Purpose

The Release Spooled File (RLSSPLF) command releases a specified spooled file formerly held on an output queue for processing by a spooling writer. The RLSSPLF command can release a spooled file that was held by:

- A HLDSPLF command
- HOLD(*YES) being specified in the device file or on an override command
- SAVE(*YES) being specified in the device file, on an override command, or in the CHGSPLFA command
- A HLDWTR command and a RLSWTR command with OPTION(*BYPASS) specified
- The operator canceling a system request to put forms on the printer

Required Parameters

FILE

Specifies the name of the spooled file that is being released to be written to an output device.

***SELECT:** All spooled files that meet the selection requirements specified in the SELECT keyword are released. This value is mutually exclusive with the JOB and SPLNBR parameters. Specifying *SELECT causes the JOB and SPLNBR parameters to be ignored.

spooled-file-name: Specify the name of the spooled file being released.

Optional Parameters

JOB

Specifies the name of the job that created the file being released for additional processing.

A job identifier is a special value or a qualified name with up to three elements. For example:

```
*
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** The job that issued this RLSSPLF command is the job that produced the spooled file being released. If no job qualifier is given, all of the jobs currently in the system are searched for the simple job name.

job-name: Specify the name of the job that created the spooled file.

user-name: Specify the name of the user of the job that created the file being released.

job-number: Specify the number of the job that created the file being released.

SPLNBR

Specifies the number of the spooled file being released.

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ONLY:** One spooled file from the job has the specified file name. The number of the spooled file is not necessary. If *ONLY is specified and more than one spooled file has the specified file name, a message is sent.

***LAST:** The spooled file with the highest number and the specified file name is used.

spooled-file-number: Specify the number of the spooled file to release that has the specified file name.

SELECT

Specifies which group of files is selected for being released. Files can be selected based on user, device, form type, or user data. Only files that meet each of the requirements are selected.

Element 1: User Values

***CURRENT:** Only files created by the user running this command are released.

***ALL:** Files created by all users are released.

user-name: Specify the user name of the files being released.

Element 2: Device Values

***ALL:** Files queued for any device or on any object queue are released.

***OUTQ:** All files that are not queued for a device are released. These files are on output queues that are not associated with printers.

device-name: Specify the name of the device whose queued files are released.

Element 3: Form Type Values

***ALL:** Files for all form types are released.

***STD:** Only files that specify the standard form type are selected.

form-type: Only files with the specified form type are released.

Element 4: User Data Values

***ALL:** Files with any user data tag specified are released.

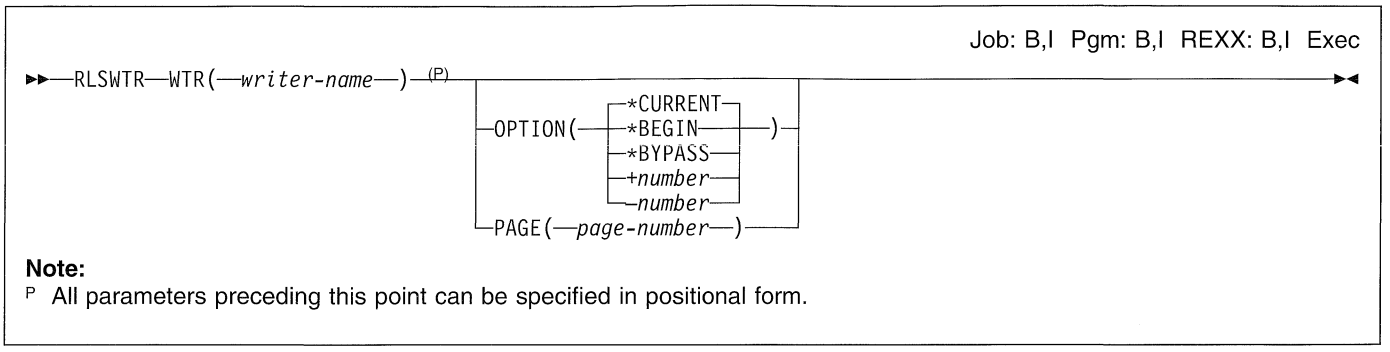
user-data: Only files with the specified user data tag are released.

Example

```
RLSSPLF  FILE(STOCK14)  JOB(000047/SMITH/MASTER)
```

This command releases the spooled file named STOCK14 created in the job named MASTER. The file can now be selected for processing by the spooling writer. The job was run under the user profile named SMITH and was assigned the job number 000047 by the system.

RLSWTR (Release Writer) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Release Writer (RLSWTR) command releases a held writer for additional processing. The specified writer was held by a previously issued HLDWTR (Hold Writer) command. If the writer was writing a file when it was held, the writer can be released either to resume writing the file it was writing or to start writing the next file. In any case, data from the file that was being written when the HLDWTR command was issued is not lost.

Required Parameters

WTR

Specifies the name of the spooling writer being released to do additional processing.

Optional Parameters

OPTION

Specifies the point in the file where the writer is released. Also, a diskette writer can be released if *CURRENT is specified.

***CURRENT:** The writer is released at the point where it was held by the HLDWTR (Hold Writer) command. This option may be specified when the writer is not producing a file.

***BEGIN:** The writer is released at the beginning of the current file. This option may be specified only if the writer was held while producing the current file.

***BYPASS:** The writer is released at the beginning of the next file. The current file is implicitly held on the queue. This option may be specified only if the writer was held while producing the current file.

+number: Specify the number of pages past the point where the writer was held that it is released. An error message is sent to the user if the specified value goes beyond the number of records or printed pages in the file. This option may be specified only if the writer was held while producing the file.

-number: Specify the number of pages preceding the point where the writer was held that it is released. An error message is sent to the user if the specified value is greater than the number of records or printed pages previously processed in the file before the writer was held. This option may be specified only if the writer was held while producing the file.

PAGE

Specifies the page where the writer starts printing. This parameter is mutually exclusive with the OPTION parameter and is only valid for a print writer. An error message is sent if the PAGE parameter is specified for a diskette writer. Specify the page number where the writer starts printing.

Examples

Example 1: Releasing a Writer at Beginning of File

```
RLSWTR  WTR(PRINTER)  OPTION(*BEGIN)
```

This command releases writer PRINTER to begin producing the current file at its beginning.

Example 2: Releasing Writer at Specified Point

```
RLSWTR  WTR(PRTR)  OPTION(-3)
```

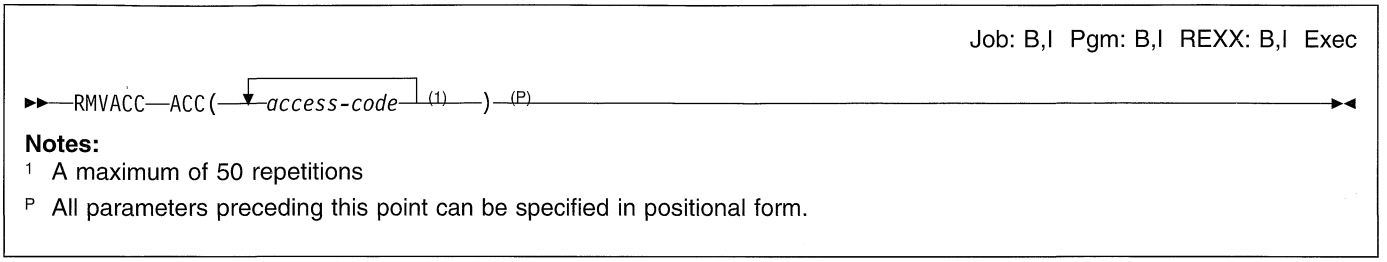
This command releases writer PRTR to begin printing again at a point three pages before the point where the writer was held. That is, the last three pages previously printed are the first three pages printed this time.

Example 3: Starting Printing on Page Ten

```
RLSWTR  WTR(PRTR)  PAGE(10)
```

This command releases writer PRTR to start printing again at page ten.

RMVACC (Remove Access Code) Command



Purpose

The Remove Access Code (RMVACC) command removes an access code from the system that was previously defined by the Add Access Code (ADDACC) command.

Notes:

1. This command can take a long time to run because it must update each object in the document library that has been assigned the access code being removed.
2. This command removes the access code from all filed documents and folders, from all users authorized to the access code, and from the system.

Restriction: This command is shipped with public *EXCLUDE authority.

Required Parameters

ACC

Specifies the access codes to remove from the system. The access code is a decimal number ranging from 1 through 2047. If the access code specified is not defined to the system, a diagnostic message is sent, and processing continues with any additional access codes specified.

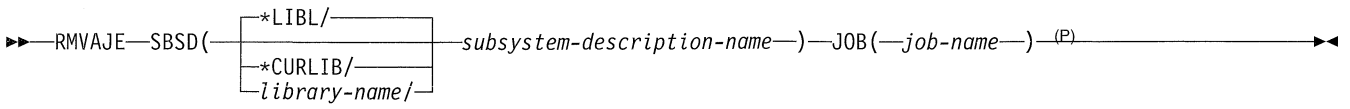
Example

```
RMVACC ACC(300)
```

This command removes access code 300 from the system.

RMVAJE (Remove Autostart Job Entry) Command

Job: B,I Pgm: B,I REXX: B,I Exec



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Autostart Job Entry (RMVAJE) command removes a job entry that starts automatically from the specified subsystem description.

Restriction: To use this command, the user must have object operational and object management authorities for the specified subsystem description.

Required Parameters

SBSD

Specifies the qualified name of the subsystem description from which the autostart job entry is being removed.

The name of the subsystem description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

subsystem-description-name: Specify the name of the subsystem description from which the autostart job entry is being removed.

JOB

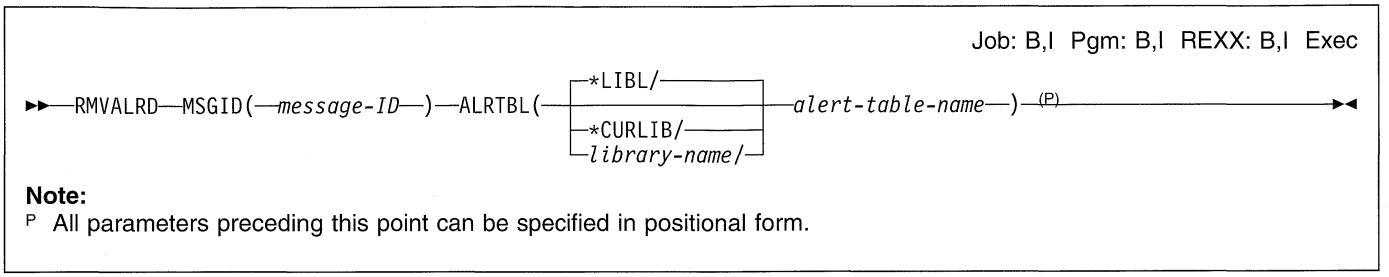
Specifies the name of the job entry to remove.

Example

```
RMVAJE  SBSD(MYLIB/PAYROLL)  JOB(INITIAL)
```

This command removes job entry named INITIAL that starts automatically from the PAYROLL subsystem description in the library MYLIB.

RMVALRD (Remove Alert Description) Command



Note:
| P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Alert Description (RMVALRD) command allows you to remove an alert description that was added previously by the ADDALRD command. More information on alerts is in the *Alerts and DSNX Guide*.

Required Parameters

MSGID
Specifies the message identifier for the alert description to be removed.

ALRTBL
Specifies the alert table from which this alert description is removed.

| The name of the alert table can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

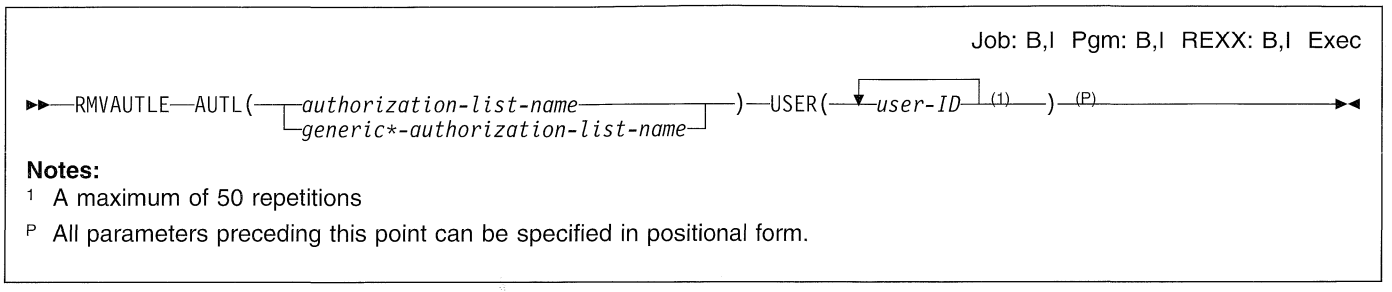
alert-table-name: Specify the name of the alert table.

Example

```
RMVALRD  MSGID(USR1234)  ALRTBL(USER/USRMSGS)
```

This command removes the alert description for message identifier USR1234.

RMVAUTLE (Remove Authorization List Entry) Command



Purpose

The Remove Authorization List Entry (RMVAUTLE) command removes user entries from an authorization list. The authorization list must already exist.

Restrictions: (1) Only the owner of the authorization list, a user with authorization list management authority (*AUTLMGT) on the authorization list, or a user with all object (*ALLOBJ) authority can use this command. (2) The user with *AUTLMGT authority can only remove a user if the user with *AUTLMGT authority has at least the same specific authorities as the user being removed.

Required Parameters

AUTL

Specifies the name of the authorization list from which the user entries are removed. The authorization list must exist.

authorization-list-name: Specify the authorization list name from which the user entries are removed.

generic-authorization-list-name:* Specify the generic name of the authorization list. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be removed only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

USER

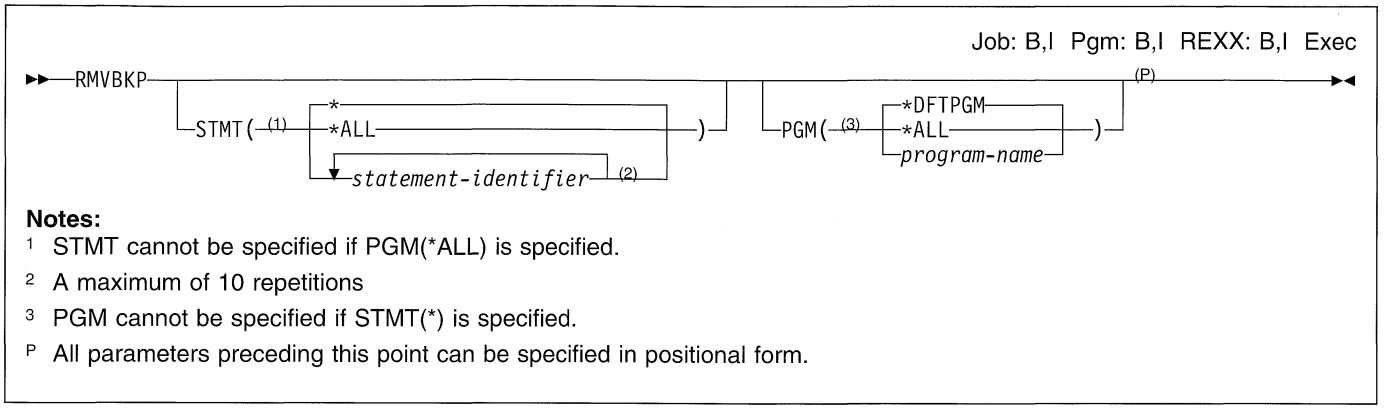
Specifies the user entry or entries to be removed from the authorization list. Up to 50 user entries can be specified.

Example

```
RMVAUTLE   AUTL(PAYROLL)  USER(TOM JULIE KAREN)
```

This command removes users TOM, JULIE, and KAREN from the authorization list PAYROLL.

RMVBKP (Remove Breakpoint) Command



Notes:

1 STMT cannot be specified if PGM(*ALL) is specified.

2 A maximum of 10 repetitions

3 PGM cannot be specified if STMT(*) is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Breakpoint (RMVBKP) command removes one or more breakpoints from the specified program being debugged. It can also remove all breakpoints from all programs in debug mode.

Restrictions:

1. This command is valid only in debug mode. To start debug mode, refer to the STRDBG (Start Debug) command.
2. This command cannot be used if the user is servicing another job, and that job is on a job queue, or is being held, suspended, or ended.
3. This command cannot be used to remove breakpoints from a bound program.

Optional Parameters

STMT

Specifies which high-level language (HLL) statements or machine instructions in a program have their breakpoints removed. Breakpoints can be removed from a specified program (PGM parameter) or from the most recent program that has reached a breakpoint (STMT(*) specified). If a program is specified, one or more statement identifiers can be specified or all the breakpoints can be specified. If STMT(*) is specified, the breakpoint that the most recently stopped program has reached is removed. Also, all breakpoints can be removed from all programs in debug mode.

***:** The most recent breakpoint at which a program is currently stopped is the breakpoint removed.

***ALL:** All breakpoints in the specified program are removed.

statement-identifier: Specify the statement identifiers removed from the program specified by the PGM parameter. No more than 10 identifiers can be specified.

PGM

Specifies the program from which the specified breakpoints are removed. This parameter can be specified only if STMT (*) is omitted.

***DFTPGM:** The default program is the program whose breakpoints are removed.

***ALL:** All programs currently in debug mode have their breakpoints removed. PGM(*ALL) is valid only if the STMT parameter is not specified.

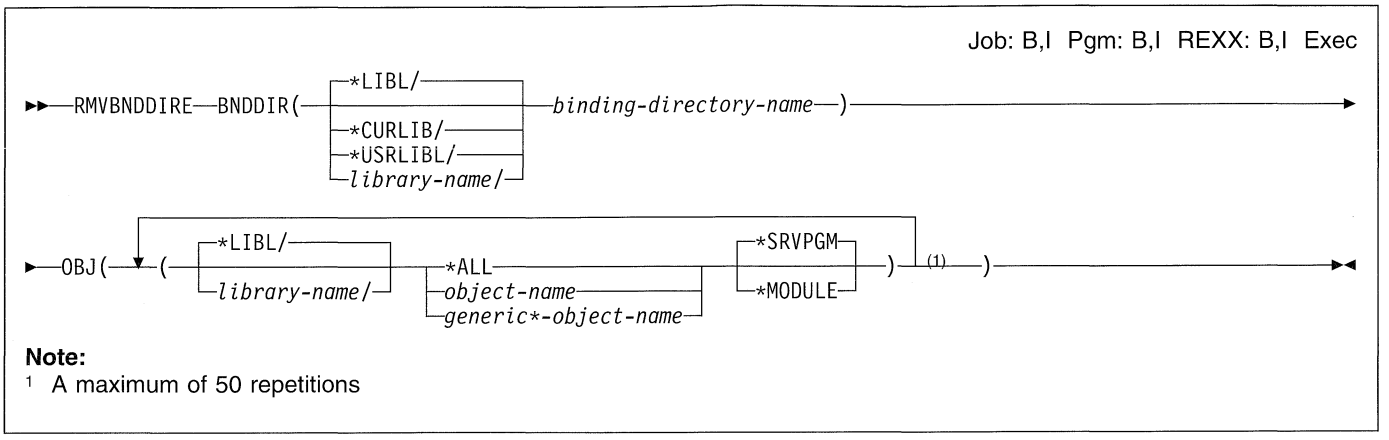
program-name: Specify the name of the program from which the specified breakpoints are removed.

Example

```
RMVBKP    STMT(100)
```

This command removes the breakpoint that is on statement 100 from the default program.

RMVBNDDIRE (Remove Binding Directory Entry) Command



Purpose

The Remove Binding Directory Entry (RMVBNDDIRE) command removes an entry from the binding directory.

Restrictions:

1. You must have *USE authority for the library where the binding directory is being updated.
2. You must have object operational and *DELETE authority to the binding directory.

Required Parameters

BNDDIR

Specifies the binding directory from which an entry is removed.

The name of the binding directory can be qualified by one of the following library values:

__*LIBL:__ All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

library-name: Specify the name of the library to be searched.

binding-directory-name: Specify the name of the binding directory to be updated.

OBJ

Specifies the object name to be removed from the binding directory.

Element 1: Removing an Object by Name

The name of the object can be qualified by one of the following library values:

__*LIBL:__ All libraries in the user and system portions of the job's library list are searched.

library-name: Specify the name of the library to be searched.

***ALL:** All objects with the specified type are removed from the specified library.

object-name: Specify the name of the object to remove.

generic-object-name:* Specify the generic name of the object. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be removed only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Element 2: Removing an Object by Type

__*SRVPGM:__ Indicates the object being removed is a service program.

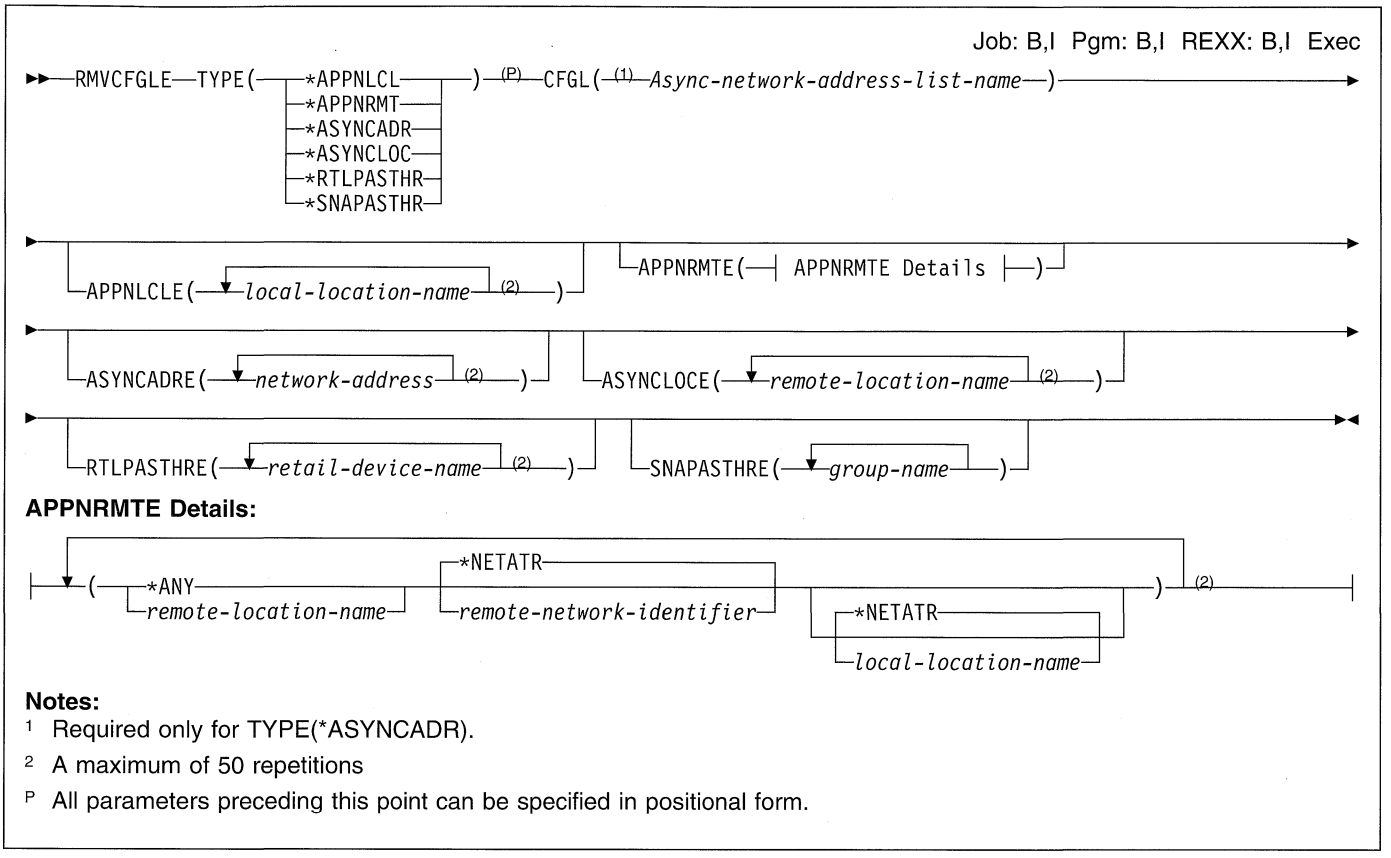
***MODULE:** Indicates the object being removed is a module.

Example

```
RMVBNDDIRE   BNDDIR(SOURCE)  OBJ(LIST)
```

This command allows you to remove the object LIST from the binding directory SOURCE.

RMVCFGLE (Remove Configuration List Entries) Command



Notes:
1 Required only for TYPE(*ASYNCADR).
2 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Configuration List Entries (RMVCFGLE) command removes entries from a configuration list.

Note: The user may also use the full screen entry display of the Change Configuration List (CHGCFGL) command to add, remove, or change entries in an existing list except for the configuration list TYPE(*SNAPASTHR).

Required Parameters

TYPE
Specifies one of the possible configuration list types.

***APPNLCL:** An APPN local location list is used. The user can specify up to 476 APPN local location entries in the configuration list.

***APPNRMT:** An APPN remote location list is used. The user can specify up to 1898 APPN remote location entries in the configuration list.

***ASYNCADR:** An asynchronous network address list is used. The user can specify up to 294 asynchronous network address entries in the configuration list.

***ASYNCLOC:** An asynchronous remote location list is used. The user can specify up to 4995 asynchronous remote location entries in the configuration list.

***RTLPASTHR:** A retail pass-through list is used. The user can key up to 450 retail pass-through entries in the configuration list.

***SNAPASTHR:** An SNA pass-through list is used. The user can key one SNA pass-through entry in the configuration list.

CFGL
Specifies the name of the configuration list. This parameter is valid only when *ASYNCADR is specified on the TYPE parameter. Only one of the other configuration list types is allowed on a system. The list types have system-supplied names: QAPPNLCL, QAPPNRMT, QASYNCADR, QASYNCLOC, QRTLPASTHR, QSNAPASTHR.

Optional Parameters

APPNLCLE
Specifies the APPN local location entry. If TYPE(*APPNLCL) is specified, this value is required. Up to 50 entries can be specified for this parameter.

APPNRMTE
Specifies the APPN remote location entry. If TYPE(*APPNRMT) is specified, this value is required. Up to 50 entries can be specified for this parameter.

Element 1: Remote Location Name

***ANY:** The system potentially accepts all requests sent to it.

remote-location-name: Specify the remote location of the entry being removed from the configuration list.

Element 2: Remote Network Identifier

***NETATR:** The RMTNETID value specified in the system network attributes is used.

remote-network-identifier: Specify the remote network identifier of the entry being removed from the configuration list.

Element 3: Local Location Name

***NETATR:** The LCLLOCNAME value specified in the system network attributes is used.

local-location-name: Specify the local location of the entry being removed from the configuration list.

ASYNCADRE

Specifies the asynchronous network address entry. This value is required if TYPE(*ASYNCADR) is specified. Up to 50 entries can be specified for this parameter.

Note: All entries having the same network address as that specified are removed from the configuration list.

ASYNCLOCE

Specifies the asynchronous remote location entry. This value is required if TYPE(*ASYNCLOC) is specified. Up to 50 entries can be specified for this parameter.

RTLPASTHRE

Specifies the retail pass-through entry. This value is required if TYPE(*RTLPASTHR) is specified. Up to 50 entries can be specified for this parameter.

Note: All entries that have the same retail device name as the one the user specified are removed from the configuration list.

SNAPASTHRE

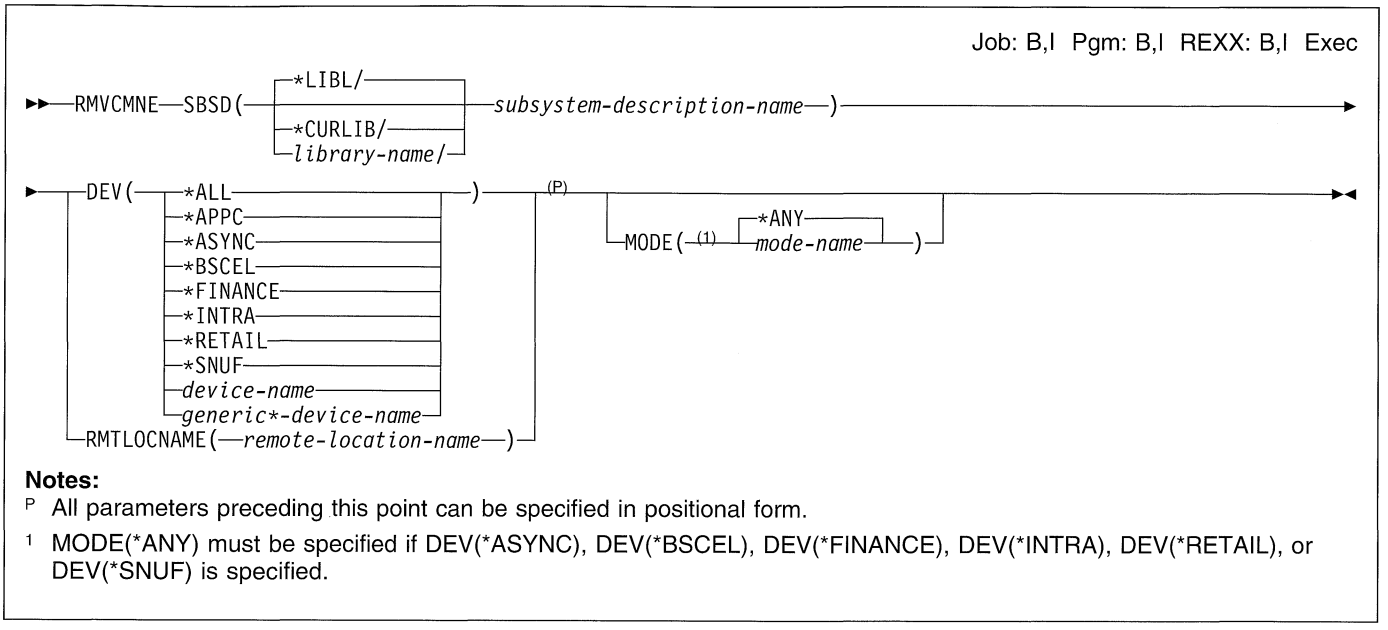
Specifies the SNA pass-through entry. This value is required if TYPE(*SNAPASTHR) is specified. One group entry can be specified for this parameter.

Example

```
RMVCFGLE  TYPE(*ASYNCLOC)  ASYNCLOCE(RMTLOC1)
```

This command removes the configuration list entry RMTLOC1 from the asynchronous remote location list QASYNCLOC.

RMVCMNE (Remove Communications Entry) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 MODE(*ANY) must be specified if DEV(*ASYNC), DEV(*BSCEL), DEV(*FINANCE), DEV(*INTRA), DEV(*RETAIL), or DEV(*SNUF) is specified.

Purpose

The Remove Communications Entry (RMVCMNE) command removes a communications device entry from an existing subsystem description. The subsystem must not be active when this command is entered.

Restriction: The user of this command must have object operational and object management authorities for the specified subsystem description.

Required Parameters

SBSD

Specifies the qualified name of the subsystem description from which the communications device entry is being removed.

The name of the subsystem description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

subsystem-description-name: Specify the name of the subsystem description.

DEV

Specifies the name of the device description or the type of the device for which a communications device entry is being removed from the subsystem description.

Note: The DEV parameter and the RMTLOCNAME parameter are mutually exclusive.

***ALL:** All communications device entries are removed with this remove communications entry.

***APPC:** The advanced program-to-program communications device entry is removed.

***ASYNC:** The asynchronous communications device entry is removed with this communications entry. This value is valid only when MODE(*ANY) is specified.

***BSCEL:** The binary synchronous equivalency link communications device entry is removed. This value is valid only when MODE(*ANY) is specified.

***FINANCE:** The FINANCE communications device entry is removed. This value is valid only when MODE(*ANY) is specified.

***INTRA:** The INTRA communications device entry is removed. This value is valid only when MODE(*ANY) is specified.

***RETAIL:** The RETAIL communications device entry is removed. This value is valid only when MODE(*ANY) is specified.

***SNUF:** The SNA upline facility communications device entry is removed. This value is valid only when MODE(*ANY) is specified.

device-name: Specify the device description name or the type of device to use with this communications device entry. The name specified on the CRTDEVxxx command associated with this device description name is used.

generic-device-name:* Specify the generic device description used with this remove communications device entry. A generic name is a character string of

one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be removed only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

RMTLOCNAME

Specifies the name of the remote location that is used with this object.

Note: The remote location name specified on the CRTDEV command can be used. Validity checking is not done on the remote location name.

Optional Parameters

MODE

Specifies the name of the mode of the device specified by the DEV parameter or the remote location specified by the RMTLOCNAME parameter from which the communications device entry is removed.

***ANY:** The communications device entry which has a value of *ANY for the device is removed.

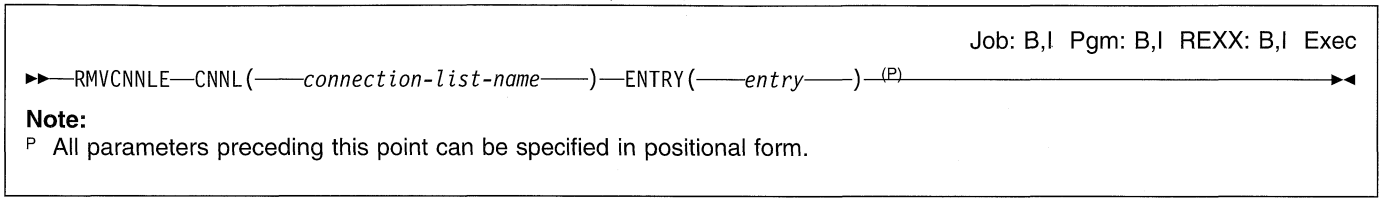
mode-name: Specify the name of the mode entry of the communications device or remote location from which the communications device entry is removed.

Example

```
RMVCMNE  SBSD(LIB2/SBS1)  DEV(COMDEV)
```

This command removes the communications device entry for the device COMDEV from the subsystem description SBS1 in library LIB2.

RMVCNNLE (Remove Connection List Entry) Command



Purpose

The Remove Connection List Entry (RMVCNNLE) command removes an entry from the specified connection list.

Required Parameters

CNNL
Specifies the name of the connection list.

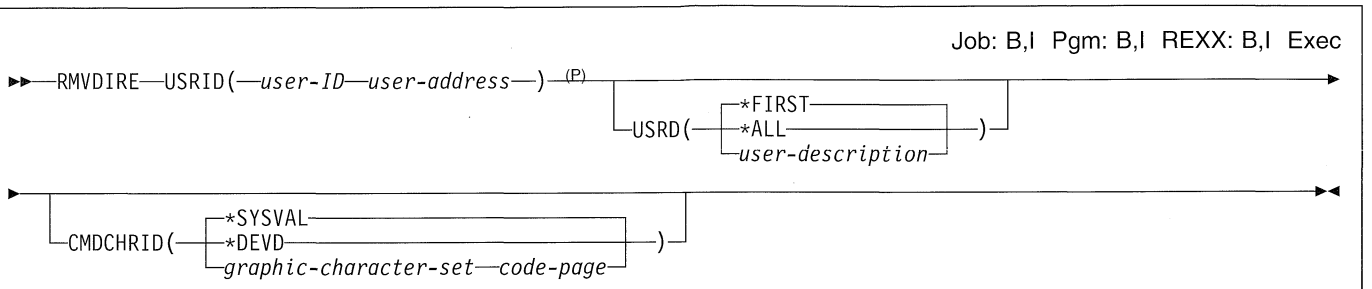
ENTRY
Specifies which entry in the connection list is removed.

Example

```
RMVCNNLE   CNNL(CHICAGO)  ENTRY(CORPORATE)
```

This command removes entry CORPORATE from the connection list CHICAGO.

RMVDIRE (Remove Directory Entry) Command



Note:

(P) All parameters preceding this point can be specified in positional form.

Purpose

The Remove Directory Entry (RMVDIRE) command removes a specific entry from the system distribution directory. When a user ID and address is removed from the directory, it is also removed from all distribution lists and the list of nicknames for that user. However, the user ID and address for that user is not removed in the list of nicknames for other users.

If an *ANY user is removed and an *ANY *ANY directory entry exists, the user is not removed from the distribution lists but the description is changed to the *ANY *ANY description.

If a user ID and address has multiple descriptions associated with it, options exist to remove only a specific description or all descriptions.

The RMVDIRE command does not provide interactive display support. This is provided by the Work with Directory (WRKDIR) command.

If the specified user ID, address, and description do not exist in the directory, an error message is returned.

Restrictions:

1. The person running this command must have security administrator (*SECADM) authority.
2. The user ID and address being removed must not:
   - Be enrolled in OfficeVision/400
   - Have ownership of documents or folders in the Document Interchange Architecture (DIA) library

   If either of the above conditions are not met, the user ID and address is not removed from the directory.
3. If the user ID and address being removed has not received mail from the mail queue, the following can happen:
   - If the request is entered interactively, a status message is sent asking whether the queue should be cleared. If the user responds yes, the mail is cleared from the queue and the user is removed from the directory. If the user responds no, the user ID and address is not removed from the directory.
   - If the request is not entered interactively (for example, the request is from a batch CL program), the user ID and address is not removed from the directory.

Required Parameter

USRID

Specifies the user ID and address for the user entry to be removed.

Element 1: User ID

user-ID: Specify the user ID that is removed. Up to 8 characters can be specified.

Element 2: User Address

user-address: Specify the user address to be removed. Up to 8 characters can be specified.

Optional Parameters

USRD

Specifies the description associated with the user ID and address. Since more than one entry can exist in the directory for a specific user ID and address, the description fully defines the user entry being removed.

***FIRST:** The first entry in the directory for the specified user ID and address is removed. If only one entry exists, it is removed.

***ALL:** All entries with the specified user ID and address are removed.

user-description: Specify up to 50 characters for the description for the user. This description must be the same as the one in the directory for this user ID and address.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify

the command. More information about CHRID processing is in the *Guide to Programming Displays*.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values used to create the command parameters. Valid values range from 1 through 9999 characters.

Element 2: Code Page

code-page: Specify the code page. Valid values range from 1 through 9999.

Example

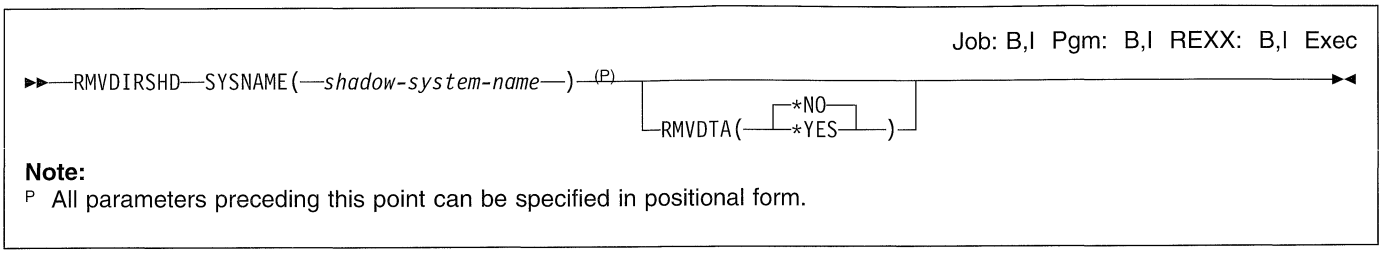
```
RMVDIRE   USRID(HURST NEWYORK)
  USRD('Manager of Payroll')
```

User ID and address HURST NEWYORK is removed if the following is true:

- An entry exists in the directory with the specified user ID, address, and description.
- The user does not own any documents or folders in the document interchange architecture (DIA) library.
- The user is not enrolled in the OfficeVision/400.
- The user has received all mail from the mail queue.

In addition, the user is removed from all distribution lists.

RMVDIRSHD (Remove Directory Shadow System) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Directory Shadow System (RMVDIRSHD) command removes a system that is currently shadowing directory data to the local system.

Restriction: To use this command, you must have security administrator (*SECADM) authority.

Required Parameters

SYSNAME

Specifies the name of the system for which shadowing is to be removed. The name can contain a maximum of eight alphanumeric characters. You can specify uppercase letters A through Z, numbers 0 through 9, and special characters @, #, $, and embedded blanks. Embedded blanks must be enclosed in single quotation marks ('). Leading blanks are not allowed. The @, #, and $ characters are not recommended because they are not part of an invariant character set and are not available on all keyboards.

Optional Parameters

RMVDTA

This parameter specifies whether to remove directory data received from the system that is being removed.

***NO:** Directory data that has been previously shadowed is left on the local system. Modifications are not made to this data through shadowing unless you shadow data from another system that has shadowed data from the system being removed.

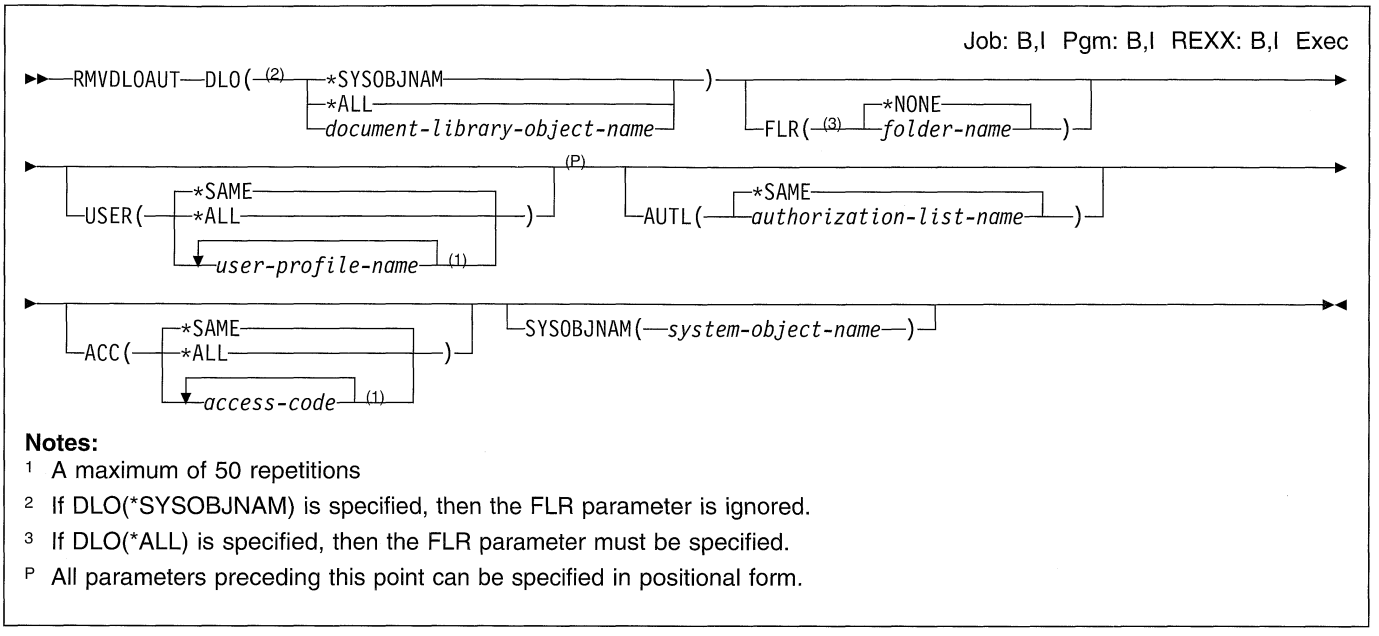
***YES:** Directory entry data that was shadowed from the specified system is removed from the local system. Department and location data is not removed.

Example

```
RMVDIRSHD   SYSNAME(NYCITY)  RMVDTA(*YES)
```

This command removes the system NYCITY from shadowing and removes all the data shadowed from NYCITY.

RMVDLOAUT (Remove Document Library Object Authority) Command



Notes:

1 A maximum of 50 repetitions

2 If DLO(*SYSOBJNAM) is specified, then the FLR parameter is ignored.

3 If DLO(*ALL) is specified, then the FLR parameter must be specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Document Library Object Authority (RMVDLOAUT) command is used to remove an existing user's authority to documents or folders.

The following types of authority can be removed:

- An existing specific user's authority
- An existing authorization list's authority to the associated document library object
- An existing object access code associated with the document library object

Restriction: The user of this command must have *ALL authority to the objects, be the owner of the objects, or have *ALLOBJ authority.

Required Parameters

DLO

Specifies the document or folder from which authority is removed.

***SYSOBJNAM:** The system object name specified in the SYSOBJNAM parameter has user authority removed.

***ALL:** All objects in a specified folder have user authority removed. If DLO(*ALL) is specified, then the FLR parameter must be specified.

document-library-object-name: Specify the user-assigned name of the document or folder from which user authority is removed. Up to 12 characters can be specified.

Optional Parameters

FLR

Specifies the name of the folder that contains the document.

***NONE:** A folder name is not specified. If DLO(name) is specified and the object is located in a folder, then FLR(*NONE) cannot be specified. If DLO(*ALL) is specified, then FLR(*NONE) cannot be specified.

folder-name: Specify the user-assigned name of the folder in which the object specified in the DLO parameter is located. The name can consist of a series of folder names if the folder containing the object is located in another folder. Up to 63 characters can be specified.

USER

Specifies the name of a user whose specific authority is removed.

***SAME:** The value does not change.

***ALL:** Specific user authority for all users is removed.

user-profile-name: Specify the name of the user profile from which specific authority is removed. Up to 50 user profile names can be specified.

AUTL

Specifies that the authority specified in the existing authorization list for the object named in the OBJ parameter is removed.

***SAME:** The value does not change.

authorization-list-name: Specify the name of the existing authorization list whose authority for the object is removed.

ACC

Specifies the access codes to be removed for the object.

***SAME:** The value does not change.

***ALL:** All access codes for the object are removed.

access-code: Specify which currently assigned access codes (ranging from 0 through 2047) are removed. Up to 50 access codes can be specified. Access code zero (0) stands for public access of *USE.

SYSOBJNAM

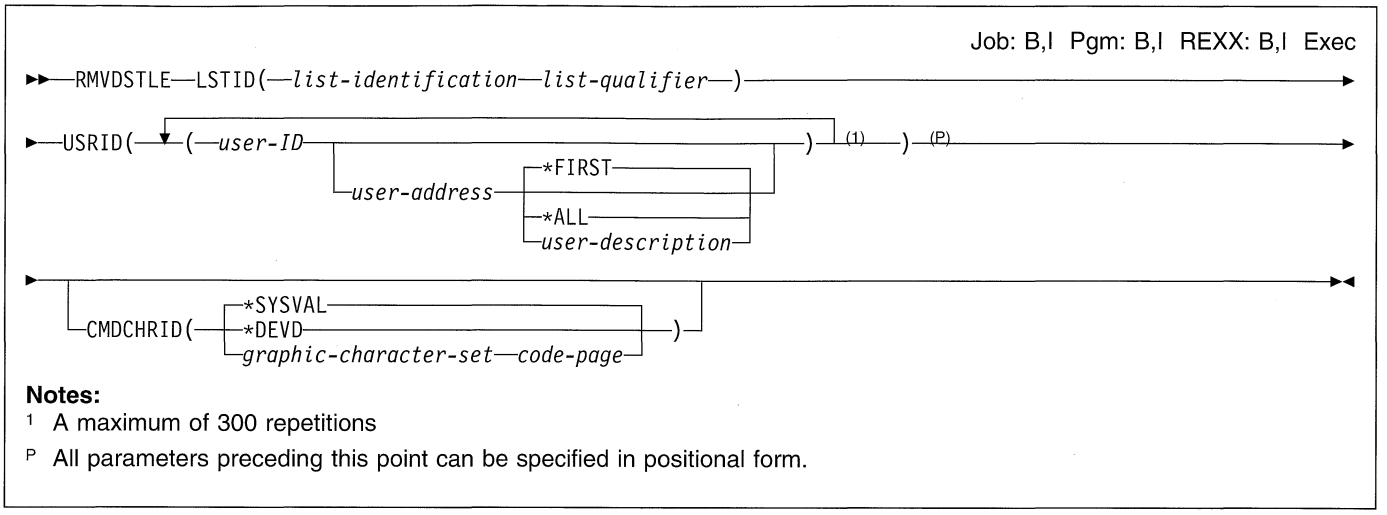
Specifies the system object name. This parameter is valid only when DLO(*SYSOBJNAM) or DOCL(*SYSOBJNAM) is specified. A full ten characters must be specified.

Example

```
RMVDLOAUT  DLO(DOCA)  FLR(MYFLR)  AUTL(MYLIST)
```

This command removes the authority of the authorization list MYLIST for object DOCA in folder MYFLR.

RMVDSTLE (Remove Distribution List Entry) Command



Notes:
1 A maximum of 300 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Distribution List Entry (RMVDSTLE) command removes entries from an existing distribution list. Up to 300 entries can be removed from a list at one time.

Restriction: The user must have security administrator authority (*SECADM) to remove entries from a distribution list owned by another user. Users can remove entries from distribution lists that they created.

Required Parameters

LSTID

Specifies the two-part list identifier of the distribution list that is to have entries removed.

Element 1: List Identifier

list-identifier: Specify the list identifier (ID) of a distribution list.

Element 2: List Qualifier

list-qualifier: Specify the list qualifier of the distribution list.

Note: The distribution list identifier/list qualifier is in two parts, separated by at least one space. If any lowercase characters are specified, the system changes them to uppercase.

The naming rules for the two-part list ID are identical to the rules for the user ID and address. A complete description of these rules is in the *Distribution Services Network Guide*.

USRID

Specifies the user ID, address, and description of the users for whom distribution list entries are removed. Both the user ID and address must be provided and must be separated by at least one space. If any lowercase characters are specified, the system translates and stores them as uppercase. The description can be entered to specify the removal of a specific user ID.

A list ID and address can be used in place of the user ID and address to identify a remote distribution list that is removed from the distribution list.

Up to 300 sets of user IDs, addresses, and descriptions can be specified. Each valid set is removed from the distribution list.

Element 1: User ID

user-ID: Specify the user ID (or the valid list ID for a remote list entry) of the user for whom the entry is removed.

Element 2: User Address

user-address: Specify the user address (or the valid list address for a remote list entry) of the user for whom the entry is removed.

Element 3: User Description

***FIRST:** The first description in the specified user ID and address (or list ID and address) is removed. If only one entry exists, it is the one removed from the list.

***ALL:** All the entries with the specified user ID and address are removed from the distribution list.

user-description: Specify the description for the user. If a list ID is specified, specify the list description. The description can be up to 50 characters in length.

Optional Parameters

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the *Guide to Programming Displays*.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values used to create the command parameters. Valid values range from 1 through 9999.

Element 2: Code Page

code-page: Specify the code page. Valid values range from 1 through 9999.

Example

```
RMVDSTLE LSTID(CHICAGO DLIST)
  USRID((HURST PAYROLL 'Manager of Payroll')
  (LEE DEPT554 *FIRST)
  (BOCA DLIST 'Remote Distribution list for Boca')
  (BRYON WAREHSE *ALL))
```

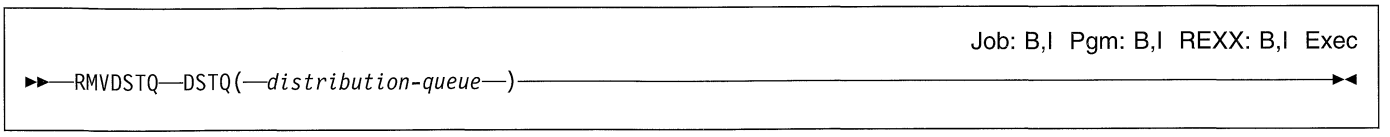
In this example, four user IDs are removed from the distribution list CHICAGO DLIST. The third user ID is, in fact, a remote distribution list. All entries for BRYON WAREHSE are removed from the list.

Additional Considerations

To remove entries from a list ID, the list ID must exist in the directory. If the list does not exist, an error message is returned.

Up to 300 sets of user IDs, addresses and user descriptions can be entered on this command. Each set of user IDs and addresses is examined to determine whether it exists in the directory. If no such entry exists, an error message is returned. If an entry is found, it is removed from the list.

RMVDSTQ (Remove Distribution Queue) Command



Purpose

The Remove Distribution Queue (RMVDSTQ) command allows the user to remove a distribution queue entry from the distribution services queue table. Distribution queues are used to store distributions before they are sent or forwarded to other systems.

The RMVDSTQ command does not provide interactive display support. This is provided by the Configure Distribution Services (CFGDSTSRV) command. More information about configuring a distribution network is in the *Distribution Services Network Guide*.

Distribution queue names are translated to the graphic character set and code page 930 500, using the job's coded character set identifier (CCSID).

Restrictions:

1. This command is shipped with public *EXCLUDE authority, and the QPGMR and QSYSOPR user profiles have private authorities to use the command.
2. The following distribution queues cannot be removed:
   - Queues referred to in the routing table
   - Queues that contain distributions waiting to be sent
   - DLS (document library services) queues that have remote libraries configured to use them
   - SVDS (SystemView distribution services) queues when a receiver is active or when distributions have been received and the sender has not acknowledged receiving confirmation.
3. Messages that report errors about distribution queues may display or print different characters than the user entered for the distribution queue name because of internal system transformations. Similarly (depending on the language used for the work station), the internal value for a distribution queue name may differ from the characters shown on the Work with Distribution Queue (WRKDSTQ) command. An error may be reported if the character-string value specified for the DSTQ parameter does not match the rules for an internal distribution queue value or if it does not match the internal value for any defined distribution queue (ignoring case differences).

Required Parameter

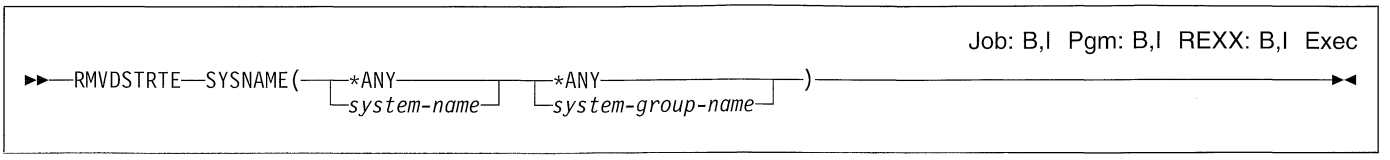
DSTQ
Specifies the name of the distribution queue entry to be removed.

Example

```
RMVDSTQ   DSTQ(CHICAGO)
```

This command removes the distribution queue entry named CHICAGO.

RMVDSTRTE (Remove Distribution Route) Command



Purpose

The Remove Distribution Route (RMVDSTRTE) command removes a remote system from the distribution services routing table. Once a system is removed from the table, distributions can no longer be sent to that system from the local system.

Interactive display support is provided by the Configure Distribution Services (CFGDSTSRV) command. More information on configuring a distribution network is in the *Distribution Services Network Guide*.

System names and system group names are translated to the graphic character set and code page 930 500, using the job's coded character set identifier (CCSID).

Restriction: This command is shipped with public *EXCLUDE authority, and the QPGMR and QSYSOPR user profiles have private authorities to use the command.

Required Parameter

SYSNAME

Specifies the system name and group name of the remote system being removed from the routing table.

Element 1: System Name

***ANY:** *ANY is used for the system name. When SYSNAME(*ANY *group*) is specified, the user removes the routing table entry used to resolve the route to a distribution destination that does not match a specific system name, but matches a group name. Only one *ANY is allowed for each group in the routing table.

system-name: Specify a maximum of 8 characters for the name of the remote system.

Element 2: System Group Name

***ANY:** *ANY is used for the system group name. *ANY can be specified for the group name only if *ANY is also specified for the system name. When SYSNAME(*ANY *ANY) is specified, the user removes the routing table entry that is used to resolve a distribution destination that does not match any other routing table entries. Only one SYSNAME(*ANY *ANY) entry is allowed in the routing table.

system-group-name: Specify a maximum of 8 characters for the system group name. The system name and group name must be separated by at least one blank.

Examples

Example 1: Removing a System from the Routing Table

```
RMVDSTRTE   SYSNAME(SYSTEMA GROUPA)
```

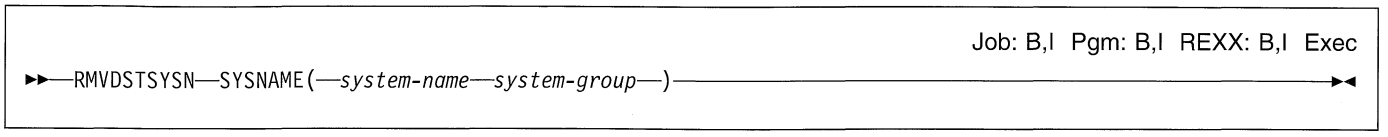
This command removes the routing table entry for the system named SYSTEMA.

Example 2: Removing a Generic Routing Table Entry

```
RMVDSTRTE   SYSNAME(*ANY GROUPNM1)
```

This command removes a routing table entry that has a system name of *ANY and a group name of GROUPNM1.

RMVDSTSYSN (Remove Distribution Secondary System Name) Command



Purpose

The Remove Distribution Secondary System Name (RMVDSTSYSN) command removes an entry from the distribution services secondary system name table. The table contains the names of all of the alternate (or alias) systems for which the local system receives distributions. When an alternate system name is removed from the table, the local system no longer receives distributions for the alternate system.

Interactive display support is provided by the Configure Distribution Services (CFGDSTSRV) command. More information about configuring a distribution network is in the *Distribution Services Network Guide*.

System names and system group names are translated to the graphic character set and code page 930 500, using the job's coded character set identifier (CCSID).

Restriction: This command is shipped with public *EXCLUDE authority, and the QPGMR or QSYSOPR user profiles have private authorities to the command.

Required Parameter

SYSNAME

Specifies the alternate system name and system group name being removed from the distribution services secondary system name table.

Element 1: System Name

system-name: Specify the name of the system for which this system is no longer to receive distributions.

Element 2: System Group Name

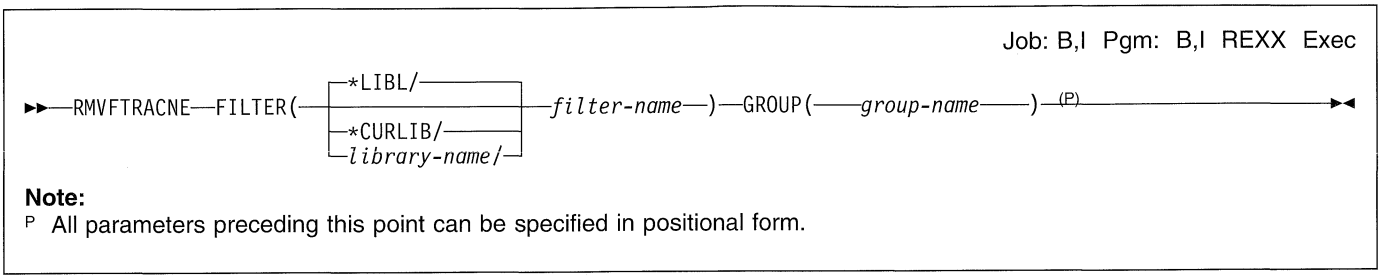
system-group: Specify the system group name of the system for which this system is no longer to receive distributions. The system name and group name must be separated by at least one blank.

Example

```
RMVDSTSYSN   SYSNAME(SYS2LAJ1 ROCHESTR)
```

This command removes the system named SYS2LAJ1 ROCHESTR from the distribution services secondary system name table.

RMVFTRACNE (Remove Filter Action Entry) Command



Note:
(P) All parameters preceding this point can be specified in positional form.

Purpose

The Remove Filter Action Entry (RMVFTRACNE) command allows the user to remove an action entry from the specified filter object.

Required Parameters

FILTER
Specifies the qualified name of the filter from which the action entry is being removed.

The name of the filter can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

filter-name: Specify the name of the filter.

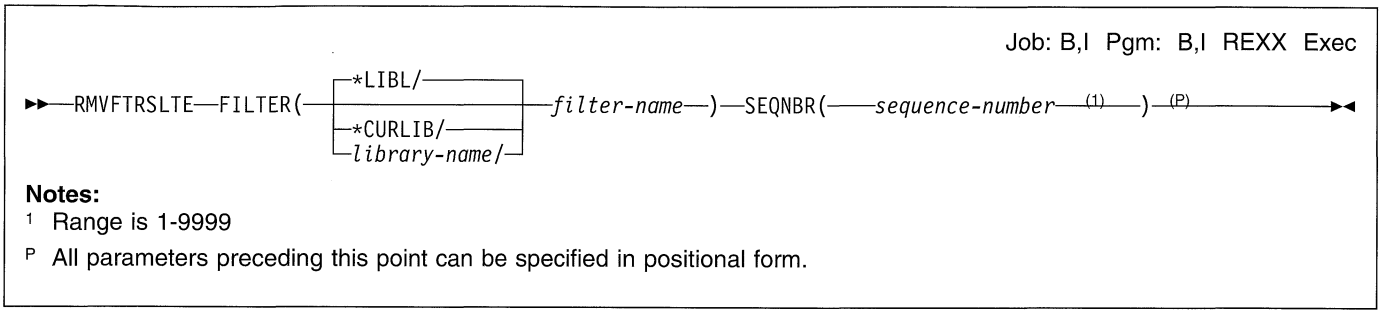
GROUP
Specifies the group that identifies the action entry being removed.

Example

```
RMVFTRACNE  FILTER(MYLIB/MYFILTER)  GROUP(CHICAGO)
```

This command removes the action entry identified by the group CHICAGO in the filter MYFILTER in library MYLIB.

RMVFTRSLTE (Remove Filter Selection Entry) Command



Notes:
1 Range is 1-9999
P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Filter Selection Entry (RMVFTRSLTE) command allows you to remove a selection entry from a filter object.

Required Parameters

FILTER
Specifies the qualified name of the filter from which the selection entry is being removed.

The name of the filter can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

filter-name: Specify the name of the filter that is used.

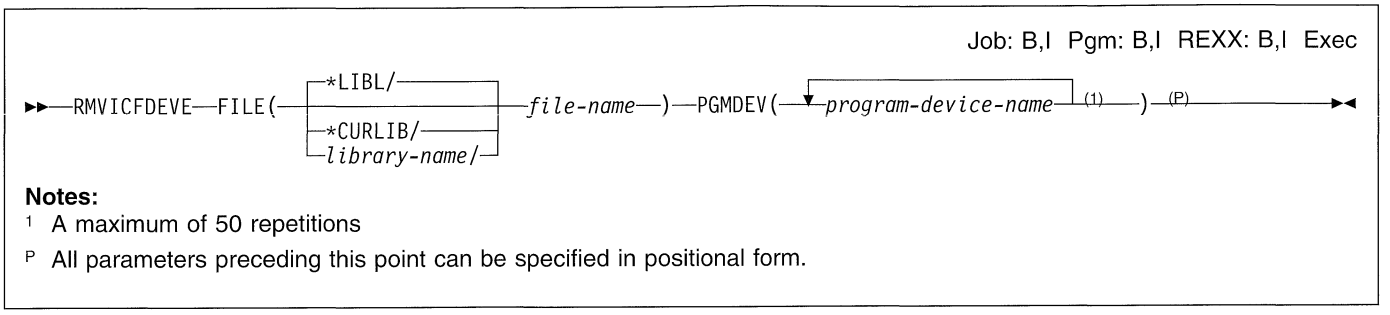
SEQNBR
Specifies the sequence number of the selection entry to be removed. Selection entries in a filter are numbered in sequence. When a filter is applied, the selection entries with the lower sequence numbers are evaluated first. Specify a number from 1 through 9999.

Example

```
RMVFTRSLTE  FILTER(MYLIB/MYFILTER)  SEQNBR(10)
```

This command removes selection entry 0010 from filter MYFILTER in library MYLIB.

RMVICFDEVE (Remove Intersystem Communications Function Program Device Entry) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Intersystem Communications Function Program Device Entry (RMVICFDEVE) command removes one or more program device entries from an ICF file.

Required Parameters

FILE

Specifies the qualified name of the ICF file from which the device entries are removed.

The name of the ICF file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the icf file from which the device entries are removed.

PGMDEV

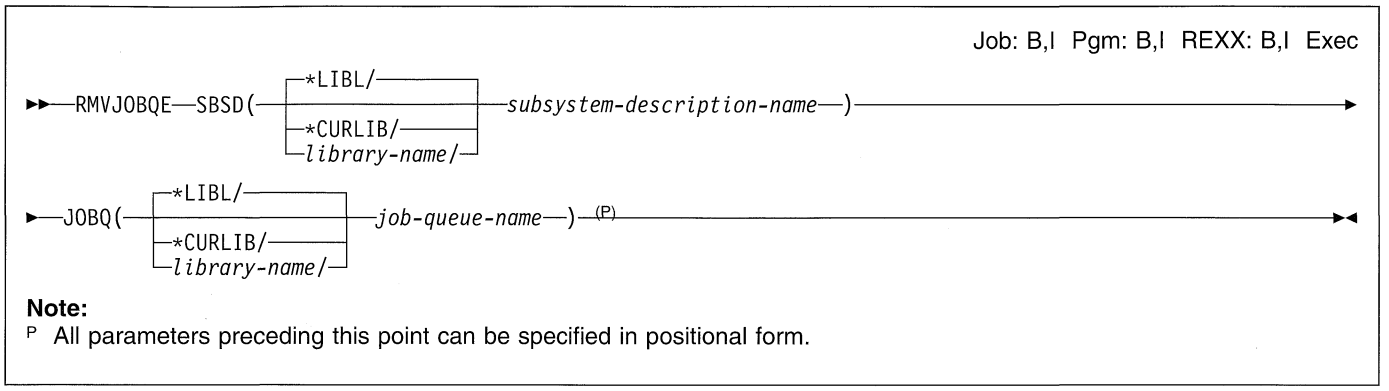
Specifies the names of up to 50 program devices that are removed from the ICF file.

Example

```
RMVICFDEVE  FILE(ICFHIST)
  PGMDEV (CHICAGO NEWYORK DENVER)
```

This command removes the program devices of CHICAGO, NEWYORK, and DENVER from the ICF file ICFHIST.

RMVJOBQE (Remove Job Queue Entry) Command



Note:

| (P) All parameters preceding this point can be specified in positional form.

Purpose

The Remove Job Queue Entry (RMVJOBQE) command removes a job queue entry from the specified subsystem description. The associated subsystem must be inactive at the time. This command can be used to remove the current job queue entry so that a different job queue can be assigned. Jobs on the queue remain on the queue for processing when the queue is reassigned to a subsystem description and the subsystem is started.

Restriction: Users of this command must have object operational and object management authorities for the specified subsystem description.

Required Parameters

SBSD

Specifies the qualified name of the subsystem description from which the job queue entry is removed.

| The name of the subsystem description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

subsystem-description-name: Specify the name of the subsystem description.

JOBQ

Specifies the qualified name of the job queue whose job queue entry is removed from the subsystem description.

| The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

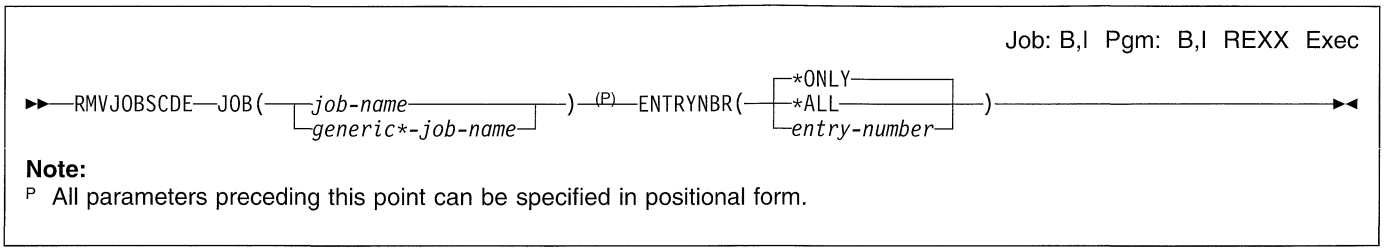
job-queue-name: Specify the name of the job queue.

Example

```
RMVJOBQE  SBSD(MYLIB/NIGHTRUN)  JOBQ(MYLIB/BATCH2)
```

This command removes the job queue entry that refers to the BATCH2 job queue in MYLIB from the NIGHTRUN subsystem description stored in library MYLIB.

RMVJOBSCDE (Remove Job Schedule Entry) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Job Schedule Entry (RMVJOBSCDE) command allows you to remove an entry, entries, or generic entries from the job schedule. Each job schedule entry contains the information needed to automatically submit a batch job one time, or at regularly scheduled intervals. A message is sent to you and to the message queue specified in the job schedule entry when an entry is successfully removed.

Restriction: To remove entries, you must have *JOBCTL special authority; otherwise you can remove only those entries that you added.

Required Parameters

JOB

Specifies the name of the job schedule entry.

job-name: Specify the name of the job schedule entry.

| *generic*-job-name:* Specify a generic name. A generic
| name is a character string of one or more characters fol-
| lowed by an asterisk (*); for example, ABC*. The
| asterisk substitutes for any valid characters. A generic
| name specifies all objects with names that begin with the
| generic prefix for which the user has authority. If an
| asterisk is not included with the generic (prefix) name,
| the system assumes it to be the complete object name.
| If the complete object name is specified, and multiple
| libraries are searched, multiple objects can be removed
| only if *ALL or *ALLUSR library values can be specified
| for the name. For more information on the use of
| generic functions, refer to "Rules for Specifying Names."
| If a generic name is specified, ENTRYNBR(*ALL) must
also be specified.

ENTRYNBR

Specifies the number of the job schedule entry you want to remove. The message sent when an entry is successfully added contains the entry number. You can also determine the entry number by using the Work with Job Schedule Entries (WRKJOBSCDE) command. Press F11 from the Work with Job Schedule Entries display to show the entry numbers of the selected entries.

***ONLY:** One entry in the job schedule has the job name specified on the JOB parameter. If *ONLY is specified and more than one entry has the specified job name, no entries are removed and a message is sent.

***ALL:** All entries with the specified job name are removed.

entry-number: Specify the number of the job schedule entry you want to remove.

Examples

Example 1: Removing Job Schedule Entries

```
RMVJOBSCDE  JOB(SAMPLE*)  ENTRYNBR(*ALL)
```

This command removes all the job schedule entries whose job names start with SAMPLE.

Example 2: Removing an Individual Job Schedule Entry

```
RMVJOBSCDE  JOB(PAYROLL)  ENTRYNBR(*ONLY)
```

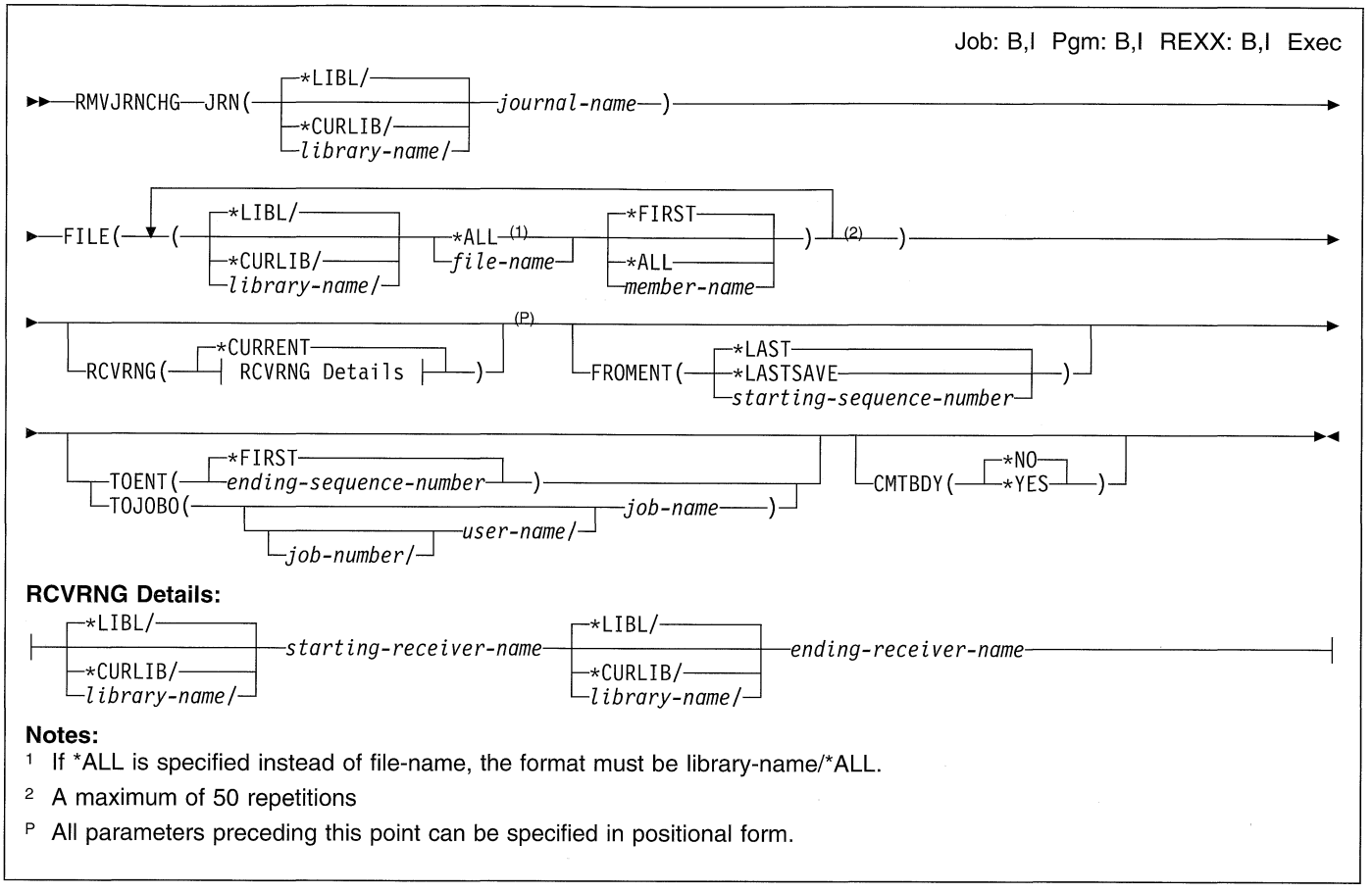
This command removes the job PAYROLL in the job schedule.

Example 3: Removing a Generic Job Schedule Entry

```
RMVJOBSCDE  JOB(PAY*)  ENTRYNBR(*ALL)
```

This command removes all entries in the job schedule that have the prefix PAY in their names.

RMVJRNCHG (Remove Journaled Changes) Command



Notes:

1 If *ALL is specified instead of file-name, the format must be library-name/*ALL.

2 A maximum of 50 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Journaled Changes (RMVJRNCHG) command removes the changes that have been journaled for a particular member of a database file. The journaled changes are removed from the file from the specified starting point to the ending point. The journal entries are processed in reverse of the order they were placed into the journal receiver, from most recent to oldest. The starting point can be identified as the last journal entry in the specified journal receiver range, the point at which a file was last saved, or a particular entry in the receiver range. The ending point can be the first journal entry or a particular entry in the specified journal receiver range, or the point at which a file was opened by a specified job. The CMTBDY parameter can be used for handling changes that are pending in the file.

Note: The Display Journal (DSPJRN) command can be used to help determine the desired starting and ending points.

A list of physical files and members may be specified. The journaled changes for all physical file members are removed in the order that the journal entries are found on the journal (the reverse of the order that the changes were originally made to the physical file members).

If an error is found at any point while the journal entries are being removed, the operation ends and the file member or members may be only partially updated from the journal entries. Errors that cause the operation to end include partial damage to a receiver and any logical error in the file member, such as a duplicate key. The command also ends when a journal entry is found that indicates one of the following has occurred:

- The member is cleared
- The member is saved and its storage is freed
- The member is reorganized
- The member is restored
- The member has its end-of-data specification changed
- Journaling has started for the member
- The member is deleted
- Journal initial program load (IPL) synchronization fails
- The system has already applied or removed the changes through the Apply Journal Changes (APYJRNCHG) command or the RMVJRNCHG command.

  The user of the command may reissue the command, specifying a new sequence starting number, if possible.

Journal entry changes can be removed even if the sequence numbers have been reset. The system handles this condition, sends an informational message, and continues the removal of journaled changes. If journal receivers are

attached and detached in pairs (dual receivers), the system initially tries to use the first of the two receivers (the first of the two receivers shown in the WRKJRNA receiver directory). When the first receiver is not usable (for example, because it is damaged or not found), the system tries to use the second receiver of the pair. If neither receiver is usable, the removal of journaled changes ends.

Restrictions:

1. This command is shipped with public *EXCLUDE authority and the QPGMR and QSRV user profiles have private authorities to use the command.
2. The files specified on this command must currently be having their changes journaled, and they must have been journaled to the specified journal throughout the period indicated on the command.
3. Before images are required for the files (see the Start Journal Physical File (STRJRNPF) command).
4. The files indicated on the command are allocated exclusively while the changes are being removed. If a file cannot be allocated exclusively, the command ends and no journaled changes are removed from the files.
5. If there is no journal entry that represents the entry specified on the FROMENT or TOENT parameter the command ends, and no journaled changes are removed from the files.
6. If the sequence number has been reset in the range of receivers specified, the first occurrence of either the FROMENT or TOENT parameter is used, if one of them is specified.

Note: If the operation ends for one of the members specified, it ends for all of the members specified.

Required Parameters

JRN

Specifies the qualified name of the journal that contains the journal entries being removed.

The name of the journal can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

journal-name: Specify the name of the journal.

FILE

Specifies a maximum of 50 qualified names of the physical database files whose journal entries are being rolled back (removed).

This parameter also specifies the name of the member in the file whose journal entries are being removed.

Element 1: File Name

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

***ALL:** All physical files in the specified library whose changes are being journaled to the specified journal have their journal entries removed. The library name must be specified. If *ALL is specified and you do not have the required authority to all of the files, an error message is sent and the command ends.

file-name: Specify the name of the physical database file whose journal entries are being removed.

Element 2: Member Name

***FIRST:** Journal entries are removed from the first member of the file.

***ALL:** Journal entries are removed from all members in the file.

member-name: Specify the name of the member in the file whose journal entries are removed.

If *ALL is specified for the first part of this parameter, the value specified for the member name is used for all applicable files in the library. For example, if *FIRST is specified, journal changes are removed from the first member of all applicable files in the library.

Note: The RMVJRNCHG command can remove changes from up to 1024 members. If more than 1024 members are specified, an error message is sent and no changes are removed. You must change the values entered on the FILE parameter.

Optional Parameters

RCVRNG

Specifies the qualified name of the first and last journal receivers used in removing the journal entries. The system begins the removal operation with the first journal receiver (specified by the first value) and proceeds through the chain of receivers until the last receiver (specified by the last value) is processed. Note that the values specified on the parameter represent journal receivers in an reverse order from that in which they were attached to the journal. If dual receivers are used at any time, the first of the receivers is always used when chaining through the set of receivers. If any problem is found in the receiver chain before the removal of journaled changes is completed (such as a damaged receiver or a receiver not found), the system tries to use the second of the dual receivers. If the

second of the receivers is damaged or not found, or if the problem is found during the removal operation, the operation ends.

***CURRENT:** The currently attached receiver is used as the only journal receiver with journal entries being removed.

Element 1: Starting Receiver Name

The name of the starting journal receiver can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

starting-receiver-name: Specify the name of the journal receiver used as the first (newest) receiver with journal entries to be removed.

Element 2: Ending Receiver Name

The name of the ending journal receiver can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

ending-receiver-name: Specify the name of the journal receiver used as the last (oldest) receiver with the journal entries being removed. If the end of the receiver chain is reached before a receiver of this name is found, the operation ends.

Note: The maximum number of receivers in the range is 256. If more receivers than this maximum are specified, an error message is sent and no changes are removed.

FROMENT

Specifies the entry used as the starting point for removing changes that were journaled.

***LAST:** Journal entries are removed starting with the last journal entry in the specified receiver range. If FROMENT is not specified, *LAST is assumed.

***LASTSAVE:** Journal entries are removed starting with the last journal entry before the last save operation. The system determines the actual starting position for each of the file members specified on the command. The parameter value implies that the file was just restored on the system.

The system also verifies that the date and time of the saved version of the file member that is restored on the system is the same as the date and time that the file member was last saved, as indicated on the journal.

If the dates and times do not match, no entries are removed and an inquiry message is sent to the system operator requesting a cancel or ignore response. If an ignore response is given to the message, the operation is attempted. A cancel response causes the operation to end, and no journal changes are removed.

If the file was last saved with the save-while-active function, the saved copy of each file member includes all record-level changes in the journal entries up to the corresponding 'F SS' start of save journal entry. In this case, the system removes changes beginning with the first journal entry preceding the 'F SS' entry.

If the file was last saved when it was not in use (normal save), the saved copy of each member includes all record-level changes in the journal entries up to the corresponding 'F MS' member saved journal entry. In this case, the system removes changes beginning with the first journal entry preceding the 'F MS' entry.

starting-sequence-number: Specify the sequence number of the first journal entry that is processed when removing journal changes from the file member.

TOENT

Specifies the entry used as the ending point for removing changes that were journaled.

***FIRST:** Journal entries are removed until the first entry in the specified receiver range is processed.

ending-sequence-number: Specify the sequence number of the last journal entry that is removed from the file member.

TOJOBO

Specifies that the journal entry changes are removed from the jobs up to and including the specified job (fully qualified job name). The specified job could be a job suspected of causing errors when the job opens a physical file member (or logical file member defined over the physical member) that is in the list of members whose journal entries are being removed, which is specified on the FILE parameter. This is the ending point for all members specified. This parameter cannot be used to remove a specific job's journal entries; all entries for all jobs are removed.

A job identifier is a qualified name with up to three elements. For example:

```
job-name
user-name/job-name
job-number/user-name/job-name
```

job-name: Specify the name of the job.

user-name: Specify the name of the user of the job.

job-number: Specify the number of the job.

CMTBDY

Specifies the method by which uncommitted changes are handled when journal changes are being applied to database files in the environment of commitment control. More information on the use of commitment control is in the *Advanced Backup and Recovery Guide*.

***NO:** The journal entries are removed from the entry specified on the FROMENT parameter to the entry specified on the TOENT parameter regardless of where the commitment boundaries are located. If the TOENT parameter identifies an entry that is in the middle of a commit cycle, the operation is attempted. The FROMENT parameter must identify a point that is at a commitment boundary.

***YES:** The journal entries are removed from the entry specified on the FROMENT parameter to the entry specified on the TOENT parameter, honoring commitment boundaries. If the specified FROMENT parameter is in the middle of a commit cycle, an error message is sent and the operation is not attempted. If the specified entry on the TOENT parameter is in the middle of a commit cycle, the operation stops at the commitment boundary that first precedes the specified entry on the TOENT parameter. When this occurs, a diagnostic message is sent at the end of the operation.

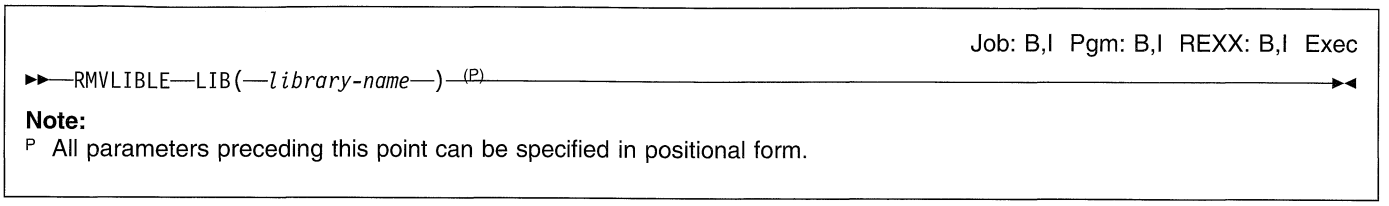
Example

```
RMVJRNCHG   JRN(JRNA)  FILE((LIB2/PAYROLL JAN))
  RCVRNG(RCV25 RCV22)  TOENT(*FIRST)
```

This command causes the system to remove all journaled changes to journal JRNA to member JAN of file PAYROLL in library LIB2 that are journaled on the journal receiver chain starting with receiver RCV25 and ending with receiver RCV22. Library search list *LIBL is used to find journal JRNA and receivers RCV25 and RCV22.

The removal operation begins with the last journaled change on the receiver chain and ends with the first journaled change.

RMVLIBLE (Remove Library List Entry) Command



Purpose

The Remove Library List Entry (RMVLIBLE) command removes a library name from the user portion of the library list.

Required Parameters

LIB

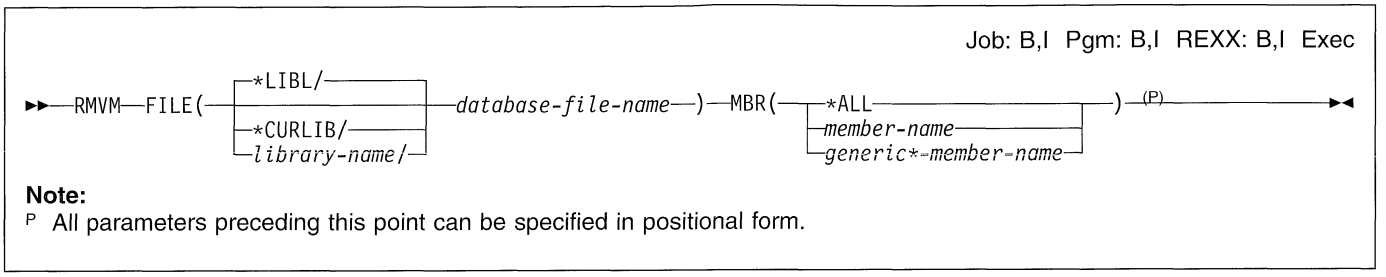
Specifies the name of the library being removed from the user portion of the library list.

Example

```
RMVLIBLE   LIB(TESTLIB)
```

This command removes the library TESTLIB from the user portion of the library list.

RMVM (Remove Member) Command



Note:

(P) All parameters preceding this point can be specified in positional form.

Purpose

The Remove Member (RMVM) command removes one or more members from the specified physical or logical file. Removing a physical file member deletes all the data in the member and frees the storage space allocated to that member and its access path. Removing a logical file member deletes its access path to the associated data stored in a physical file member.

Restriction: If a member of another file is sharing the data of the member being deleted, the dependent member must be removed first. To remove a member from a file, the user must have object existence authority for the file.

Required Parameters

FILE

Specifies the qualified name of the database file (physical or logical) that contains the member being removed.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file that contains the member being removed.

MBR

Specifies the name of the physical or logical file member being removed. A specific or generic member name, or *ALL, can be requested.

***ALL:** All members found in the specified file are removed.

member-name: Specify the specific name of the member removed.

generic-member-name:* Specify the generic name of the member. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be removed only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Examples

Example 1: Removing a File Member

```
RMVM   FILE(JOBHIST1)  MBR(JOBHIST1A)
```

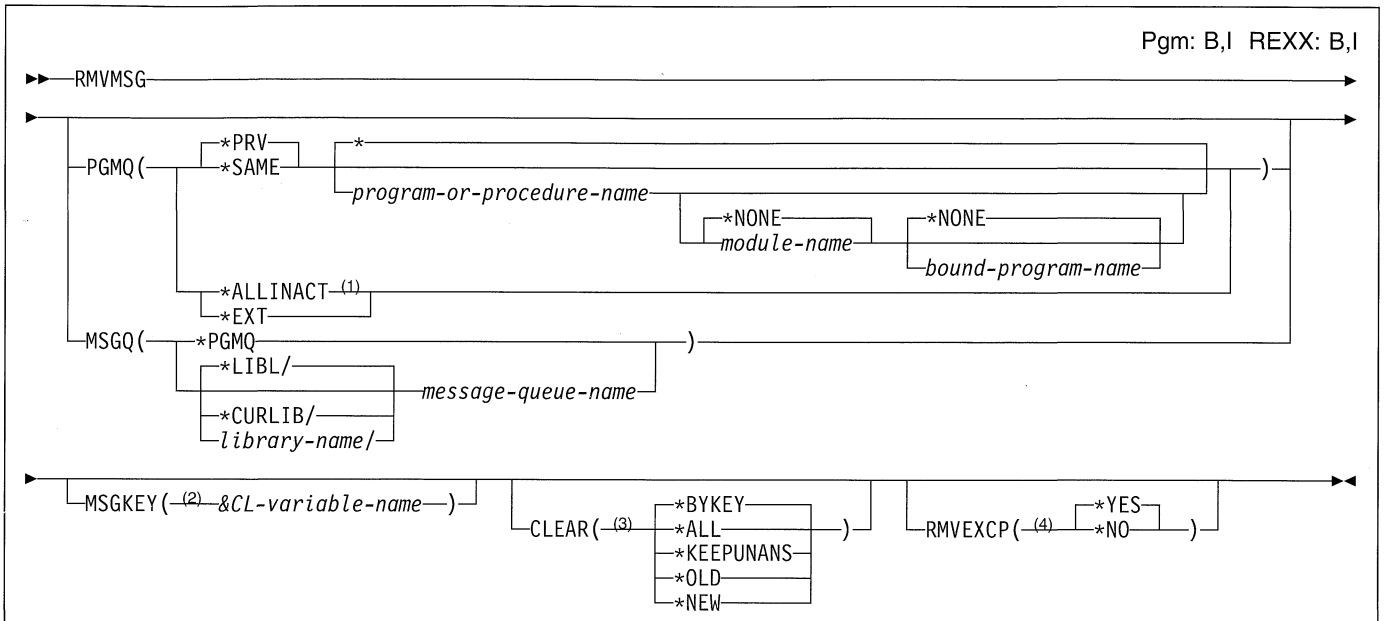
This command removes file member JOBHIST1A from file JOBHIST1. Library list *LIBL is used to find the file and member. If JOBHIST1 contains other members, they remain unchanged.

Example 2: Removing Members with Names that Start with SRC

```
RMVM   FILE(QGPL/JOBHISTL) MBR(SRC*)
```

This command removes all file members with names that start with SRC from file JOBHISTL in library QGPL.

RMVMSG (Remove Message) Command



Notes:

1 If PGMQ(*ALLINACT) is specified, the CLEAR parameter must be *ALL.

2 MSGKEY can be specified only if CLEAR(*BYKEY) is specified.

3 CLEAR(*KEEPUNANS) is valid only if a message queue name is specified for the MSGQ parameter. CLEAR(*KEEPUNANS) is not valid if *PGMQ is specified for MSGQ.

4 This parameter is valid only when MSGQ(*PGMQ) is specified.

Purpose

The Remove Message (RMVMSG) command is used by a program to remove the specified message, or a group of messages, from the specified message queue. If the specified message queue is not allocated to the job in which this command is issued, or to any other job, it is allocated by this command for the duration of the command.

Restrictions:

1. This command is valid only in a compiled CL program.
2. To remove a message from the message queue, the user must have *CHANGE authority for the queue and *USE authority for the library in which the queue is stored.

Optional Parameters

PGMQ

Specifies the call message queue from which the message is removed. Messages can be removed from the external queue (*EXT) or from a message queue associated with a call stack entry.

Note: If CLEAR(*BYKEY) is specified, the PGMQ parameter is ignored.

Element 1: Relationship

Element 1 of this parameter specifies whether the message queue is associated with the program or procedure identified by Element 2, or if it is associated with the caller of the program or procedure.

***PRV:** The message is removed from the message queue of the program or procedure that called the program or procedure identified by Element 2.

***SAME:** The message is removed from the message queue of the program or procedure identified by Element 2.

Element 2: Program or Qualified Procedure

Element 2 of this parameter has three items. Item 1 specifies the program or procedure of the job message queue. Items 2 and 3 specify the module name and the bound program name, which can be used to qualify the procedure name.

Item 1: Program or Procedure Name

***:** Identifies the program or procedure running the RMVMSG command.

program-or-procedure-name: Specify the name of the program or procedure used.

If Item 1 identifies a program, the name specified can be a maximum of 10 characters. If Item 1 identifies a pro-

cedure, the name specified can be a maximum of 256 characters. The system begins its search for the specified program or procedure name with the most recently called program or procedure.

The procedure name alone may not identify the correct procedure. Several different procedures with the same name can run in a job. To further identify a procedure, the name specified can be qualified by a module name, or by both a module name and a bound program name.

Item 2: Module Name

The module name qualifier identifies the module into which the procedure was compiled.

***NONE:** No module name is specified.

module-name: Specify the module name to be used as a qualifier for the specified procedure name (modules are associated only with procedures). The module name can be a maximum of 10 characters.

If a module name is not specified but a bound program name is, *NONE must be specified in the module name position.

Item 3: Bound Program Name

The bound program name qualifier identifies the program to which the procedure was bound.

***NONE:** No bound program name is specified.

bound-program-name: Specify the bound program name to be used as a qualifier for the specified procedure name (and module name if specified). The bound program name can be a maximum of 10 characters.

Other Single Values

***ALLINACT:** All messages for all inactive call stack entries are removed from the user's job message queue. If this value is specified for the PGMQ parameter, the value *ALL must be specified on the CLEAR parameter.

***EXT:** The message is removed from the external message queue of the job.

MSGQ

Specifies the qualified name of the message queue from which one or more messages are removed. If MSGQ is specified, the PGMQ parameter cannot be specified.

***PGMQ:** The call message queue specified in the PGMQ parameter is the only queue from which the messages are removed. If CLEAR(*KEEPUNANS) is specified, MSGQ(*PGMQ) cannot be specified.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue from which one or more messages are removed.

MSGKEY

Specifies the name of the CL variable that contains the message reference key of the message being removed. MSGKEY cannot be specified if *ALL, *KEEPUNANS, *OLD, or *NEW is specified for the CLEAR parameter.

CLEAR

Specifies whether one message, all messages (except unanswered inquiry messages), or only old or new messages in the specified message queue are removed from the queue.

***BYKEY:** The message identified by the CL variable named in the MSGKEY parameter is removed from the message queue.

***ALL:** All messages are removed from the specified message queue.

***KEEPUNANS:** Messages other than unanswered inquiry messages are removed from the specified message queue. If CLEAR(*KEEPUNANS) is specified, MSGQ(*PGMQ) cannot be specified (a message-queue name must be specified for MSGQ).

***OLD:** Old messages in the specified message queue are removed from the queue.

***NEW:** New messages in the specified message queue are removed from the queue.

RMVEXCP

Specifies the action to be taken when an unhandled exception message is found. An unhandled exception message is an escape, notify, or status message that has been sent to an ILE procedure. When this command is run, the ILE procedure has not yet taken action to tell the system that the exception is handled. One action that the ILE procedure can take is to call a CL program that will remove the exception message. More information on actions that an ILE procedure can take to handle an exception is in *ILE* Concepts*.

This parameter is valid only when working with a message queue that is associated with a call stack entry for an ILE procedure. This parameter is ignored when working with a message queue associated with a call stack entry for a program.

***YES:** The unhandled exception message on the specified message queue is removed. As a result, the exception is handled.

***NO:** The unhandled exception message on the specified message queue is not removed. The message remains on the queue as an unhandled exception message.

Examples

Example 1: Removing a Message

```
RMVMSG  MSGQ(SMITH)  MSGKEY(&KEY)
```

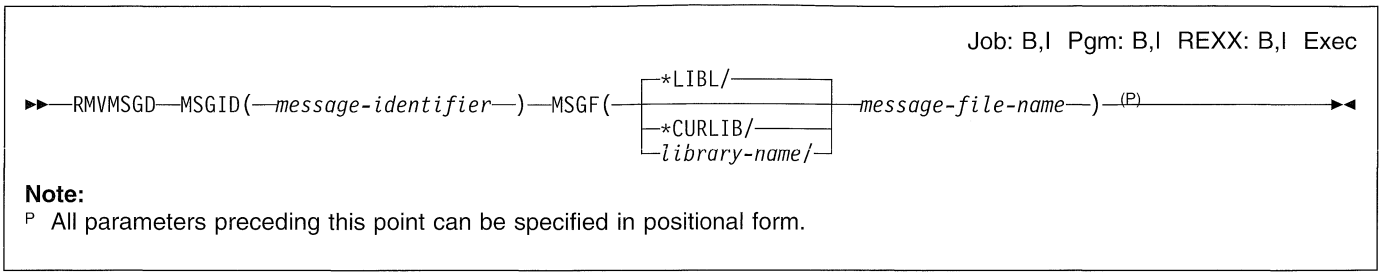
This command removes the message with the reference key specified in the CL variable &KEY from the message queue named SMITH.

Example 2: Keeping Unanswered Messages

```
RMVMSG  MSGQ(SMITH)  CLEAR(*KEEPUNANS)
```

This command removes all messages except the unanswered inquiry messages from the message queue named SMITH.

RMVMSGD (Remove Message Description) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Message Description (RMVMSGD) command removes a message description from the specified message file.

Note: A description of how to print a single message description or a group of message descriptions is in the section entitled "Handling Messages" in the *System Operator's Guide*, SC41-8082.

Restriction: The user must have use and delete authorities to the message file.

Required Parameters

MSGID
Specifies the message identifier of the message being removed from the message file.

MSGF
Specifies the qualified name of the message file containing the message being removed. Any message file overrides in effect for the job are ignored by this command; the file specified here is the one from which the message is removed.

The name of the message file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

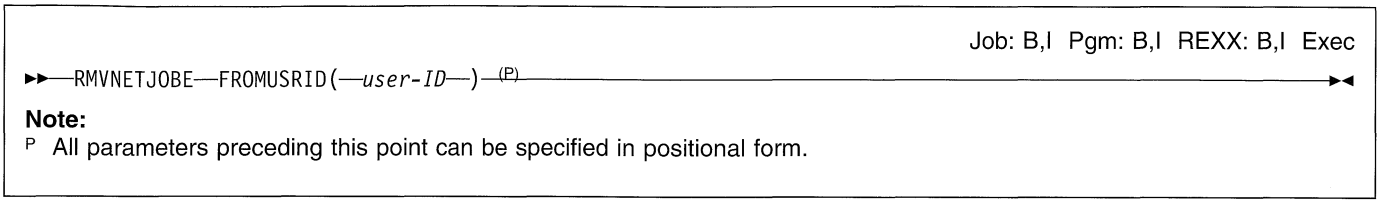
message-file-name: Specify the name of the message file to use.

Example

```
RMVMSGD   MSGID(UIN0115)  MSGF(INV)
```

This command removes the message description with the identifier UIN0115 from the message file named INV. The library list is used to find the INV file. Note that if more than one INV message file exists in the libraries being searched, the message description will only be removed from the first INV message file found in the library list.

RMVNETJOBE (Remove Network Job Entry) Command



Purpose

The Remove Network Job Entry (RMVNETJOBE) command removes a network job entry from the system. The network job entry determines the action taken when an input stream is sent to a user on this system by using the Submit Network Job (SBMNETJOB) command. This entry is used to determine whether the input stream is automatically submitted, placed on the queue of received files for a user, or rejected. The entry also specifies the user profile that is used for checking the authority to the job description referenced in the batch job. There should be one entry for each user or group of users who submit jobs to this system.

If this command is used to remove an entry for a specific user, an entry may still exist that is in effect for that user. For example, if the user removes the entry for user ID, JOE PGMRS, and if there is an entry with a user ID, *ANY PGMRS or *ANY *ANY, that entry is used to handle any jobs submitted by JOE PGMRS. Additional information on the job entry table is in the *Distribution Services Network Guide*.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. The internal value for a node identifier may differ from the characters shown by the RMVNETJOBE command depending on the type of work station (language) being used. If the byte-string value specified for the FROMUSRID command parameter does not match the rules for an internal node identifier value, or if it does not match the internal value for any defined node (ignoring case differences), an error may be reported.

Required Parameters

FROMUSRID

Specifies the two-part user ID that identifies the network job entry being removed. Specify the two-part user ID. Both parts of the user ID are required.

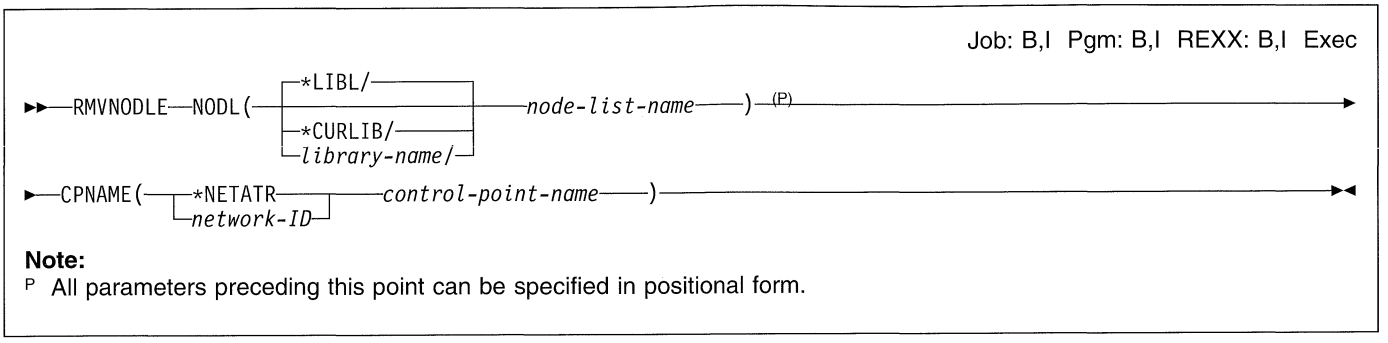
Note: Depending on the work station being used, the internal value for a new user identifier may differ from the characters shown by the Display Network Job Entry (DSPNETJOBE) command. If the byte-string value specified for the FROMUSRID parameter does not match the rules for an internal user identifier value, or if it does not match the internal value for any enrolled user, an error may be reported.

Example

```
RMVNETJOBE  FROMUSRID(JOE SMITH)
```

This command removes the network job entry that is used to determine the action that is taken for any input streams received from user ID, JOE SMITH. The network job authority for user ID, JOE SMITH, is taken from either the network job entry *ANY SMITH, if that entry exists, or from the network job entry *ANY *ANY, if that entry exists. If neither of these entries exist, all jobs received from user ID, JOE SMITH, are rejected.

RMVNODLE (Remove Node List Entry) Command



Purpose

The Remove Node List Entry (RMVNODLE) command removes an entry from an existing node list object.

Required Parameters

NODL

Specifies the qualified name of the node list object from which the entry is removed.

The name of the node list can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

node-list-name: Specify the name of the node list from which the entry is removed.

CPNAME

Specifies the system to remove from the node list object. The system is specified by its network ID and control point name.

***NETATR:** The NETID network attribute is used as the value of the network ID of the system being removed from the node list.

network-ID: Specify the network ID of the system being removed from the node list.

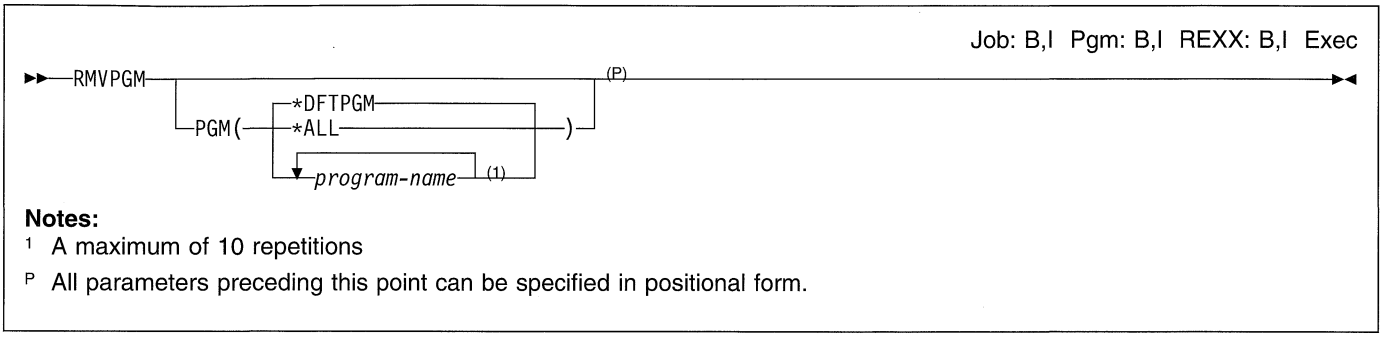
control-point-name: Specify the control point name of the system being removed from the node list.

Example

```
RMVNODLE    NODL(MYLIB/NODL02)
            CPNAME(*NETATR AS400A01)
```

This command removes the entry for system AS400A01 in the local network from node list NODL02 in library MYLIB.

RMVPGM (Remove Program) Command



Notes:
1 A maximum of 10 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Program (RMVPGM) command removes one or more programs from the current debugging session. All breakpoints and data traces defined in each program are removed, and the programs are returned to their normal state. If a program is added again, breakpoints and traces must be specified again.

Restrictions:

1. This command is valid only in the debug mode. To start the debug mode, refer to the STRDBG (Start Debug) command.
2. This command cannot be used to remove bound programs from a debugging session.

Optional Parameter

PGM

Specifies which programs are removed from the current debugging session.

***DFTPGM:** The program currently specified as the default program in the debugging session is removed. The debugging session no longer has a default program unless one is specified later.

***ALL:** All programs currently in the debug mode are removed.

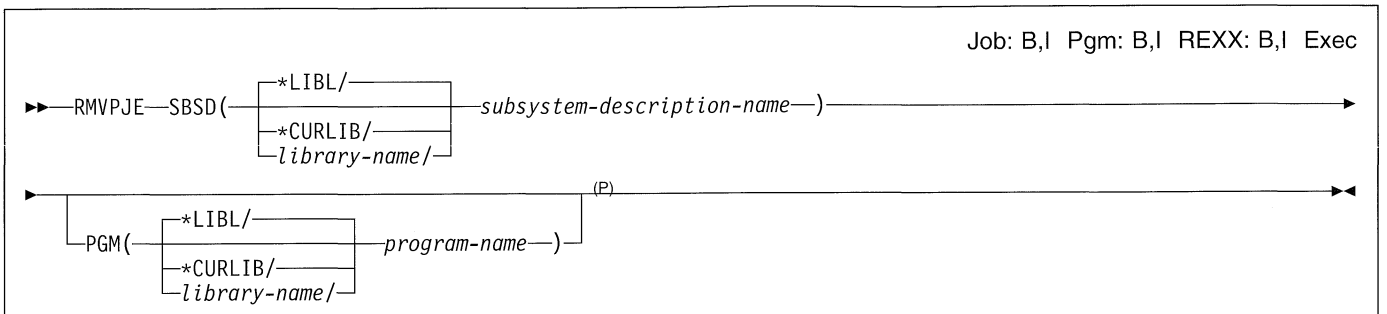
program-name: Specify the names of up to 10 programs being removed from the current debugging session.

Example

```
RMVPGM  PGM(PGMX  PGMY  PGMZ)
```

This command removes the three programs PGMX, PGMY, and PGMZ from the current debugging session. All breakpoints and data traces are removed from the programs.

RMVPJE (Remove Prestart Job Entry) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Prestart Job Entry (RMVPJE) command removes a prestart job entry from the specified subsystem description. The subsystem must be inactive when the prestart job entry is removed.

When removing an entry in which *LIBL is specified for the library name, the library list is searched for a program with the specified name. If a program is found in the library list but an entry exists with a different library name, which is found later in the library list, no entry is removed. If a program is not found in the library list but an entry exists, no entry is removed.

Restriction: This command is restricted to a user with *USE and object management authorities for the subsystem description.

Required Parameters

SBSD

Specifies the qualified name of the subsystem description that contains the prestart job entry being removed.

| The name of the subsystem description can be qualified by one of the following library values:

__*LIBL:__ All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

subsystem-description-name: Specify the name of the subsystem description.

Optional Parameters

PGM

Specifies the qualified name of the program for the prestart job entry being removed. Two entries with the same program name can exist in a single subsystem description but they must have different library names.

| The name of the program can be qualified by one of the following library values:

__*LIBL:__ All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

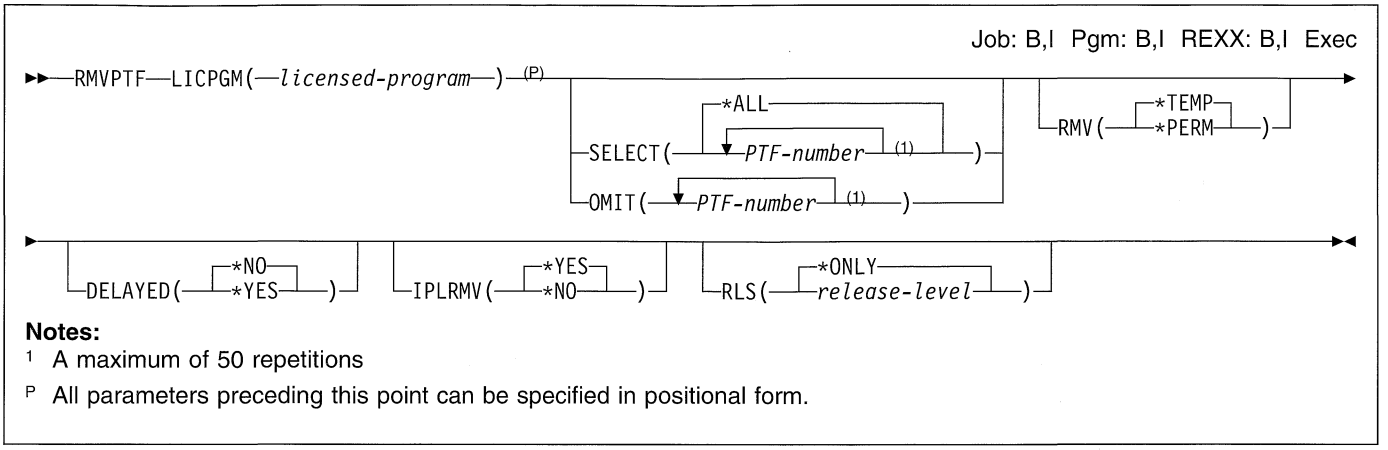
program-name: Specify the name of the program.

Example

```
RMVPJE  SBSD(QGPL/PJE)  PGM(QGPL/PGM1)
```

This command removes the prestart job entry for the PGM1 program (in the QGPL library) from the PJE subsystem description contained in the QGPL library.

RMVPTF (Remove Program Temporary Fix) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Program Temporary Fix (RMVPTF) command removes the specified program temporary fixes (PTFs) from the specified licensed program. If the PTFs were temporarily applied, the original objects that they replaced are returned. The PTFs can be temporarily removed, in which case they are held in the licensed program base library and they can be applied later. If the PTFs have not been applied, they can be permanently removed and moved to the QRPLOBJ library.

The RMVPTF command is used to remove immediate PTFs at the time the command is processed, or when the user wants to request that PTFs be removed during an unattended initial program load (IPL).

During an attended IPL, the Work with PTFs display is used to remove PTFs when the system is started.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR user profile has private authority to use the command.

Required Parameter

LICPGM
Specifies the licensed program from which the PTFs are removed. Specify the licensed program.

Optional Parameters

SELECT
Specifies which of the PTFs to remove from the specified licensed program. The OMIT parameter cannot be specified if single PTF numbers are specified in the SELECT parameter.

***ALL:** All the PTFs are removed from the licensed program. Those that were permanently applied are ignored by this command. If all PTFs cannot be removed, messages are sent to the job log indicating which PTFs are not removed and the reasons they are not being removed.

PTF-number: Specify the PTF identification number of each PTF being removed. Up to 50 PTF numbers can be specified.

OMIT
Specifies that all PTFs are removed except for those specified in this parameter. Specify the PTF numbers of the PTFs that are omitted (left in the system) when all the rest are removed. Up to 50 PTF numbers can be specified. The OMIT parameter cannot be specified if single PTF numbers are specified in the SELECT parameter.

RMV
Specifies whether the PTFs are removed temporarily or permanently. Permanently removed PTFs are deleted from the system. Temporarily removed PTFs are held in the licensed program base library for application at a later time.

***TEMP:** The PTFs are removed and held in the licensed program base library so that they can be applied again later, if desired.

***PERM:** The PTFs are permanently removed and deleted from the library.

DELAYED
Specifies whether immediate PTFs are removed at the time the command is executed, or whether immediate or delayed PTFs are removed during the next unattended IPL.

***NO:** Immediate PTFs that are identified are removed at the time the command is processed. Delayed PTFs are ignored during the RMVPTF request and are not removed.

***YES:** Both delayed and immediate PTFs that are identified are removed during the next unattended IPL. The IPLAPY parameter in the APYPTF command determines whether the PTFs are removed during the next unat-

tended IPL, or whether a request to remove the PTFs during the next unattended IPL is canceled.

IPLRMV

Specifies the action that is done for delayed or immediate PTFs at the next unattended IPL. This parameter is valid only if DELAYED(*YES) is also specified.

***YES:** The identified PTFs are removed at the next unattended IPL. The RMV parameter determines whether the remove is temporary or permanent.

***NO:** Any previous request to remove the identified PTFs at the next unattended IPL is canceled.

RLS

Specifies the release level of the PTFs being removed.

***ALL:** All PTFs are removed.

release-level: Specify the release level in the format VxRxMx, where Vx is the version number, Rx is the release number, and Mx is the modification number.

Examples

Example 1: Temporarily Removing PTFs

```
RMVPTF  LICPGM(5738SS1)  DELAYED(*YES)
```

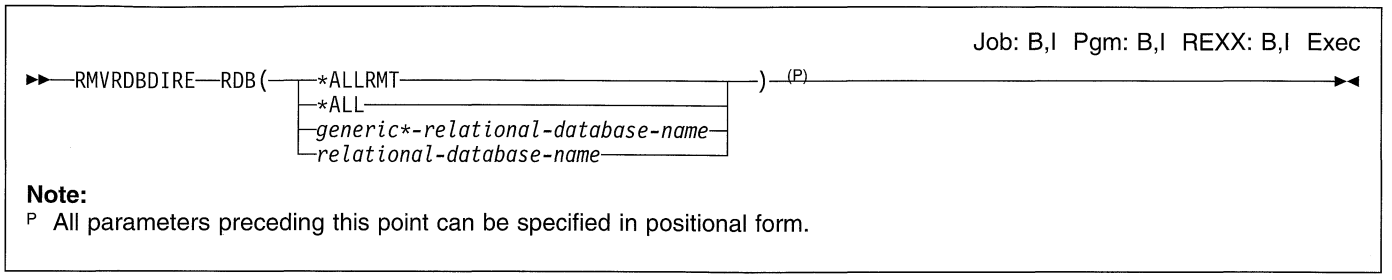
This command temporarily removes all temporarily applied PTFs from Operating System/400 (program identifier 5738-SS1). The PTFs can be applied again, if necessary, through the APYPTF command.

Example 2: Permanently Removing PTFs

```
RMVPTF  LICPGM(5738RG1)  SELECT(L100002 L100005)
  RMV(*PERM)
```

This command permanently removes two PTFs (numbers L100002 and L100005) from the RPG (licensed program 5738-RG1). The two PTFs are deleted from the system and must be loaded again using the LODPTF command before they can be applied.

RMVRDBDIRE (Remove Relational Database Directory Entry) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Relational Database Directory Entry (RMVRDBDIRE) command removes a specific entry, generic entries, all entries, or all remote entries from the relational database directory.

Required Parameters

RDB

Specifies the relational database directory entry being removed.

***ALLRMT:** All remote entries are removed from the relational database directory. The entry that is specified as RMTLOCNAME(*LOCAL) on the Add Relational Database Directory Entry (ADDRDBDIRE) command is not removed.

***ALL:** All entries in the relational database directory are removed.

generic-relational-database-name:* Specify the generic name of the relational database. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be removed only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

relational-database-name: Specify the name of the relational database being removed. Up to 18 characters can be specified.

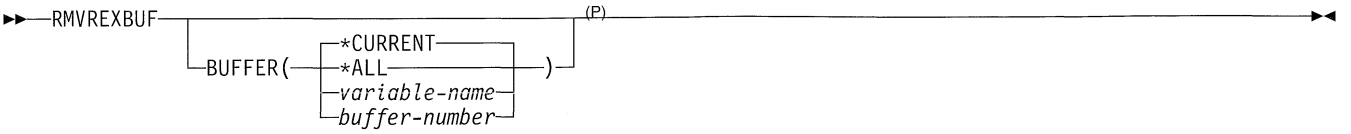
Example

```
RMVRDBDIRE   RDB(YOURRDB)
```

This command removes the entry YOURRDB from the relational database directory. The entry is no longer accessible.

RMVREXBUF (Remove REXX Buffer) Command

Job: B,I Pgm: B,I REXX: B,I Exec



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Remove REXX Buffer (RMVREXBUF) command allows the user to remove a buffer from the REXX external data queue.

Optional Parameters

BUFFER

Specifies the number of the buffer being removed. The buffer identified by the number and all buffers above it, in chronological order up to and including the current buffer, are removed.

***CURRENT:** Only the current buffer is removed.

***ALL:** All buffers and entries are removed from the REXX external data queue. This is equivalent to BUFFER(0).

variable-name: Specify the name of the variable containing the buffer number. The user must specify a decimal variable with a minimum length of 11 digits with no decimal position.

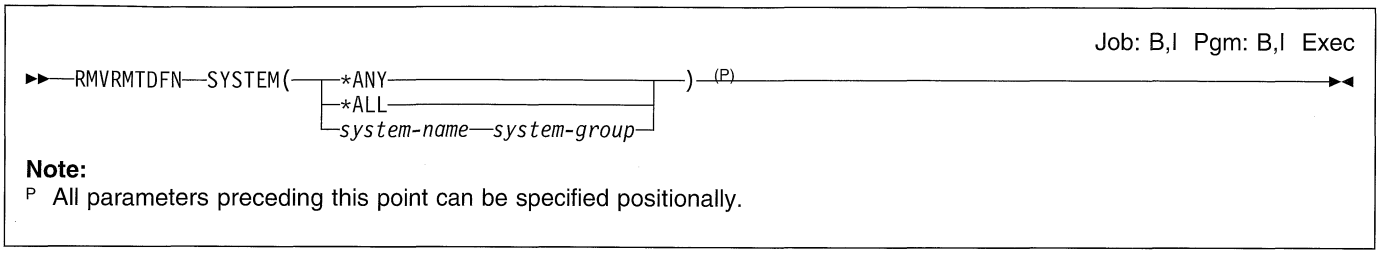
buffer-number: Specify the number of the buffer to be removed.

Example

```
RMVREXBUF  BUFFER(2)
```

This command removes buffer number 2 and all buffers with numbers higher than 2 from the REXX external data queue.

RMVRMTDFN (Remove Remote Definition) Command



Note:
P All parameters preceding this point can be specified positionally.

Purpose

The Remove Remote Definition (RMVRMTDFN) command is used to remove the definition of attributes for a remote system.

Restriction: The user must have *ALLOBJ authority.

Required Parameters

SYSTEM

Specifies the system name and system group of the remote system being removed.

***ANY:** Removes the default definition for a remote system not covered by the other entries.

***ALL:** Removes the definitions for all remote systems.

Element 1: System Name

system-name: Specify the name of the remote system being removed.

Element 2: System Group

system-group: Specify the group name of the remote system being removed. The group name is blank if this value is not specified.

Example

Example 1: Removing a Specific Remote Definition

```
RMVRMTDFN  SYSTEM(RCHAS1)
```

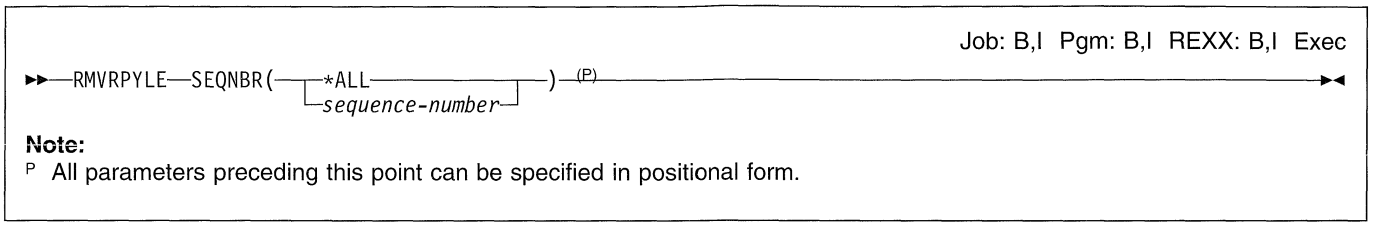
This command removes the definition from remote system RCHAS1. This system now uses the values for the *ANY remote definition or the defaults.

Example 2: Removing all Remote Definitions

```
RMVRMTDFN  SYSTEM(*ALL)
```

This command removes all remote system definitions. All systems now use the default values.

RMVRPYLE (Remove Reply List Entry) Command



Note:

(P) All parameters preceding this point can be specified in positional form.

Purpose

The Remove Reply List Entry (RMVRPYLE) command removes an entry from the system reply list. The reply list is used as a source for automatic responses to predefined inquiry messages.

The reply list is only used when an inquiry message is sent by a job that has the inquiry message reply attribute of the system reply list specified (INQMSGRPY(*SYSRPLY) is specified). INQMSGRPY(*SYSRPLY) can be changed with the CHGJOB command.

New entries may be added to the reply list with the Add Reply List Entry (ADDRPYLE) command; entries can be changed with the Change Reply List Entry (CHGRPYLE) command. The entire list of entries can be shown with the Work with System Reply List Entries (WRKRPYLE) display. From the display that is presented the user can add, change, and remove individual entries.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR user profile has private authority to use the command.

Required Parameters

SEQNBR

Specifies the sequence number of the reply list entry being removed from the system reply list.

***ALL:** Specifies that all reply list entries are removed from the system reply list. Whenever *ALL has been specified, and the reply list object had previously been marked as damaged, the reply list is deleted and recreated. No reply list entries will be present after re-creation. They must be reentered or a reinstall operation must be done.

sequence-number: Specify the sequence number of the file.

Examples

Example 1: Removing All Entries

```
RMVRPYLE   SEQNBR(*ALL)
```

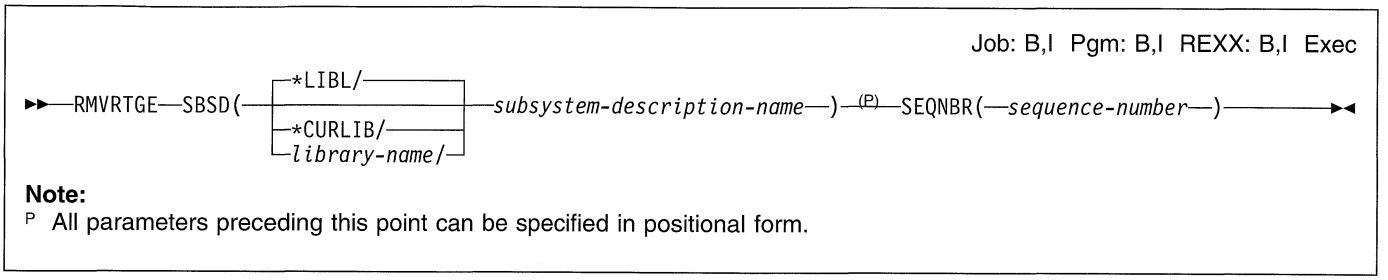
This command removes all entries from the system reply list.

Example 2: Removing One Entry

```
RMVRPYLE   SEQNBR(0001)
```

This command removes from the system reply list the entry that has sequence number 0001.

RMVRTGE (Remove Routing Entry) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Routing Entry (RMVRTGE) command is used during system set-up, when work is divided on the system, to remove a specified subsystem description routing entry that controls programs or run attributes for a job. The subsystem must be inactive at the time.

Restriction: The user, typically a system operator or system administrator, must have both object operational and object management authorities to change the subsystem description.

Required Parameters

SBSD

Specifies the qualified name of the subsystem description that contains the routing entry being removed.

| The name of the subsystem description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

subsystem-description-name: Specify the name of the subsystem description.

SEQNBR

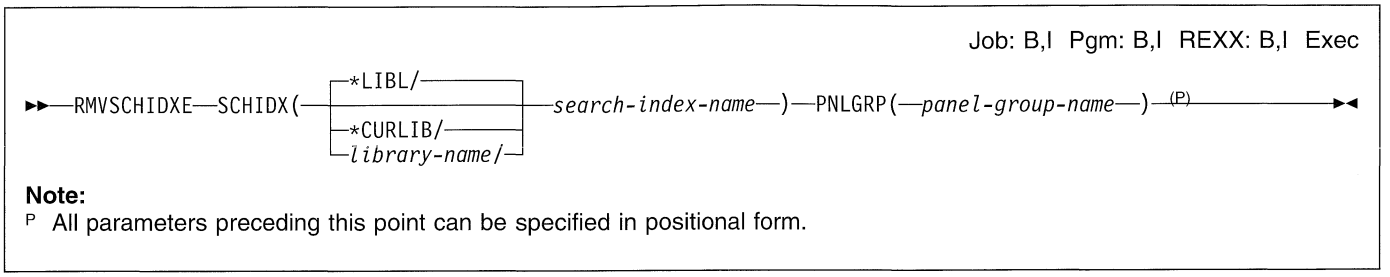
Specifies the sequence number of the routing entry being removed. The sequence numbers on the routing entries indicate the order in which the routing entries will be processed when comparing values. When a job enters the system, the compare value for the job is compared to the compare value for the routing entries in a subsystem. When the first match is found, the routing information for that entry is used for the new job; therefore, the sequence numbers indicate the compare order.

Example

```
RMVRTGE  SBSD(OR/PERT)  SEQNBR(9912)
```

This command removes the routing entry 9912 from subsystem description PERT in library OR.

RMVSCHIDXE (Remove Search Index Entry) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Search Index Entry (RMVSCHIDXE) command is used to remove entries that refer to a specified panel group from a search index.

Restrictions: The user must have *CHANGE authority for the search index.

Required Parameters

SCHIDX

Specifies the qualified name of the search index from which the entries are being removed.

The name of the search index can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

search-index-name: Specify the name of the search index.

PNLGRP

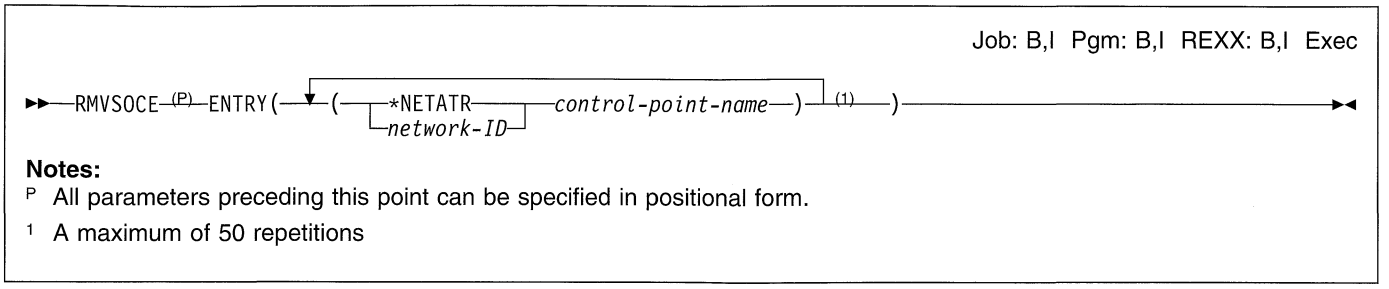
Specifies the name of the panel group for which entries are being removed.

Example

```
RMVSCHIDX   SCHIDX(ACCOUNTING)  PNLGRP(PAYROLL)
```

This command removes panel group PAYROLL from search index ACCOUNTING.

RMVSOCE (Remove Sphere of Control Entry) Command



Purpose

The Remove Sphere of Control Entry (RMVSOCE) command allows a CL user or program to remove control points from the Alert Sphere of Control.

Required Parameters

ENTRY

Specifies the systems to remove from the alert sphere of control. The systems are specified as a list of two elements which include the network ID and control point name.

Element 1: Network ID of the System

***NETATR:** The NETID network attribute is used as the value of the network ID to be removed from the alert sphere of control.

network-ID: Specify the network ID of system to be removed from the alert sphere of control.

Element 2: Control Point Name of the System

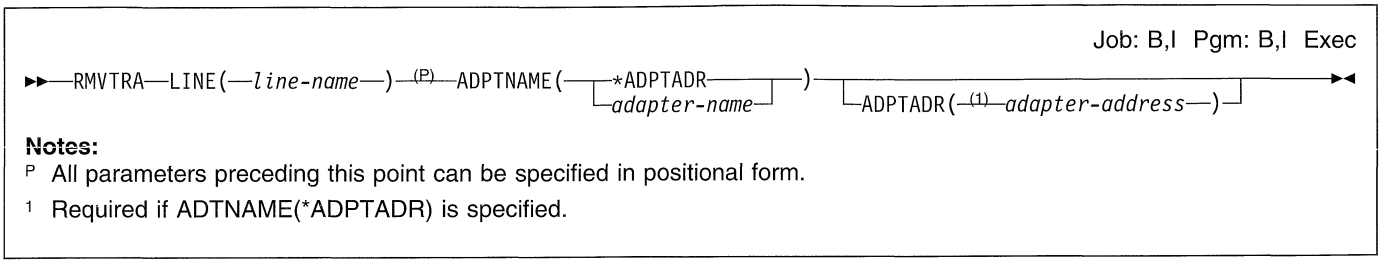
control-point-name: Specify the control point name of the system to be removed from the alert sphere of control.

Example

```
RMVSOCE  ENTRY((*NETATR RCHSTR1) (*NETATR RCHSTR2))
```

This command removes two systems (RCHSTR1 and RCHSTR2) from the alert sphere of control.

RMVTRA (Remove TRLAN Adapter) Command



Purpose

The Remove Token-Ring Local Area Network Adapter (RMVTRA) command removes an active TRLAN (token-ring local area network) adapter from a token-ring line description that is varied on.

Restrictions:

- This command is only valid for users with QSECOFR authority.
- This command is only valid for TRLAN (token-ring local area network) managers that are in the controlling mode.

  Note: The mode (controlling or observing) of the TRLAN manager is set on the TRNMGRMODE parameter when the line is created or changed with the CRTLINTRN or CHGLINTRN commands.

Required Parameters

LINE
Specifies the name of the line description attached to the adapter being removed.

ADPTNAME
Specifies the name of the adapter being removed.

***ADPTADR:** The adapter address is used to identify the adapter.

adapter-name: Specify the name of the adapter being removed.

Optional Parameters

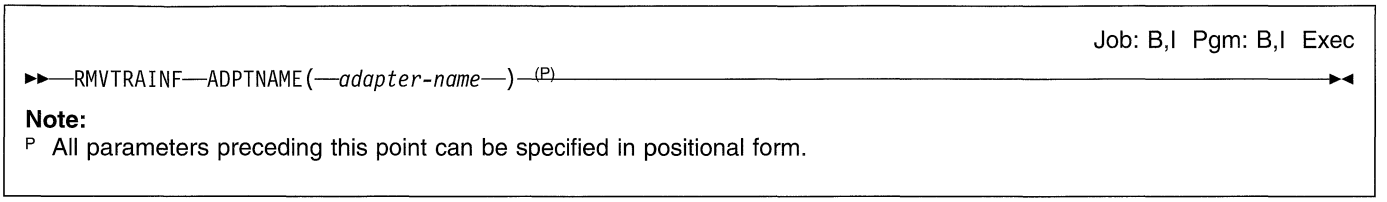
ADPTADR
Specifies the 12-character hexadecimal adapter address.

Example

```
RMVTRA  LINE(CHGBRANCH)  ADPTNAME(*ADPTADR)
        ADPTADR(000000001BFF)
```

This command removes the adapter with an address of 000000001BFF from the token-ring line description.

RMVTRAINF (Remove TRLAN Adapter Information) Command



Purpose

The Remove Token-Ring Local Area Network Adapter Information (RMVTRAINF) command removes an adapter name entry from the adapter file.

Required Parameters

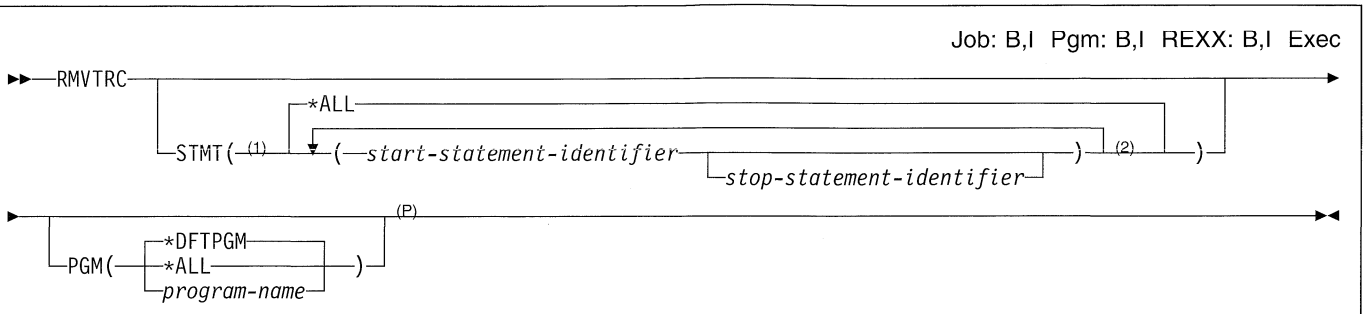
ADPTNAME
Specifies the name of the adapter being removed from the adapter file.

Example

```
RMVTRAINF  ADPTNAME(PAYROLL)
```

This command removes the adapter PAYROLL from the adapter file.

RMVTRC (Remove Trace) Command



Notes:

1 STMT cannot be specified if PGM(*ALL) is specified.

2 A maximum of 5 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Trace (RMVTRC) command removes all or part of the traces previously specified in one or more Add Trace (ADDTRC) commands for use in debugging the programs. Any trace data already created by the traces being removed is not affected by this command. (This data can be removed by the Clear Trace Data (CLRTRCDTA) command.) The tracing limits specified in the Change Debug (CHGDBG) command are not changed.

On the RMVTRC command, the user specifies the high-level-language (HLL) statement identifiers or the machine instruction numbers that correspond to the ranges that the user no longer wants traced. To remove a trace, exactly the same range (as specified on the ADDTRC command) must be specified. Up to five sets of trace ranges can be specified in one command.

Restrictions:

1. This command is valid only in debug mode. To start debug mode, refer to the STRDBG (Start Debug) command.
2. This command cannot be used if the user is servicing another job, and that job is on a job queue, or is being held, suspended, or ended.
3. This command cannot be used to remove a trace from a bound program.

Optional Parameters

STMT

Specifies the HLL statement identifiers or machine instruction numbers of the trace statements that are no longer being traced. Unless *ALL is specified, to remove a trace from a program, the same statement identifiers must be specified here that were specified on the ADDTRC command.

The method used to specify the trace statements on the ADDTRC command (that is, HLL statement identifiers versus machine instruction numbers) must also be used here to remove them.

***ALL:** All HLL statements and/or machine instructions in the specified programs are no longer traced regardless of how the trace was defined by the ADDTRC command.

Element 1: Statement Identifier of First Trace Statement

start-statement-identifier: Specify the HLL statement identifier (or the machine instruction number) of the first trace statement being removed from future tracing. Up to five trace ranges can be specified in the program for each use of this command.

Element 2: Statement Identifier of Last Trace Statement

stop-statement-identifier: Specify, optionally, the HLL statement identifier (or the machine instruction number) of the last statement being removed from future tracing. If the last statement was specified on the ADDTRC command, the last statement must also be specified here. Up to five trace ranges can be specified in the program for each use of this command.

PGM

Specifies the program (or all programs) containing the trace statements being removed from future tracing operations.

***DFTPGM:** The program currently specified as the default program contains the statements being removed from tracing.

***ALL:** All programs that currently have trace ranges in them have *all* of their trace ranges removed; no tracing can be done in any of the programs in debug mode unless more traces are added by the ADDTRC command. PGM(*ALL) is valid only if the STMT parameter is not specified.

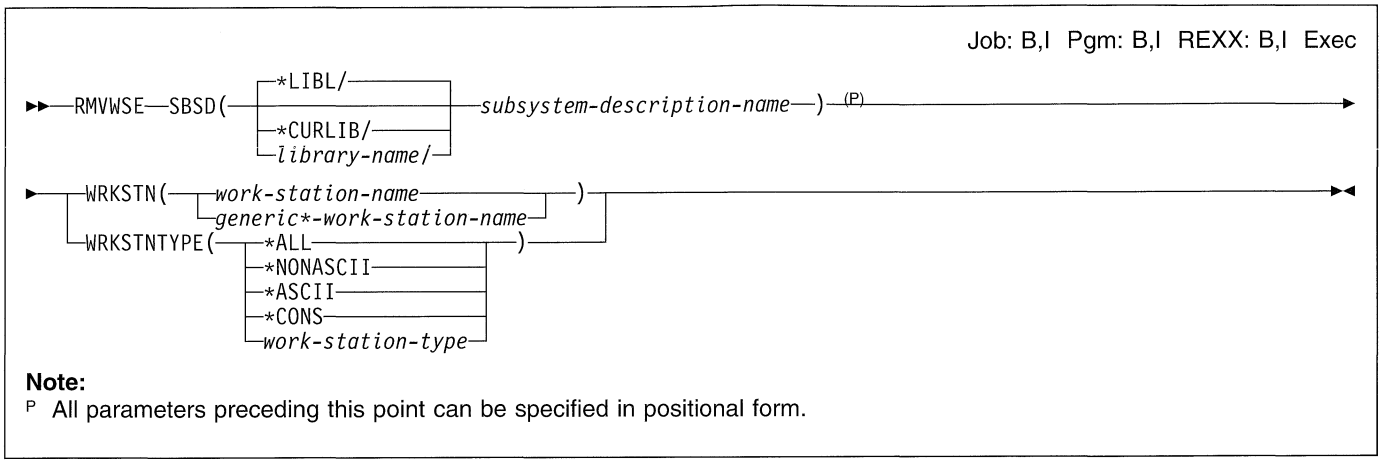
program-name: Specify the name of the program that has the specified trace statements (or all trace statements) being removed.

Example

```
RMVTRC
```

This command removes all the trace statements used for tracing in the program currently specified as the default program.

RMVWSE (Remove Work Station Entry) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Remove Work Station Entry (RMVWSE) command is used, when the system is being set up, to remove a specified work station or type of work station from a specified subsystem description. The associated subsystem must be inactive at the time.

Restriction: The user, typically a system operator or system administrator, must have operational or object management authority to change the subsystem description.

Required Parameters

SBSD

Specifies the qualified name of the subsystem description containing the work station job entry that is being removed.

The name of the subsystem description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

subsystem-description-name: Specify the name of the subsystem description containing the work station entry that is being removed.

WRKSTN

Specifies the device description name of the work station for which the job entry is being removed.

work-station-name: Specify the name of the work station whose work station entry is removed.

generic-work-station-name:* Specify the generic name of the work station. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be removed only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

The WRKSTN parameter and the WRKSTNTYPE parameter are mutually exclusive. When a generic name is specified, only the generic name entry is removed. For more information on the use of generic functions, see "Rules for Specifying Names."

WRKSTNTYPE

Specifies the work station device type for which the job entry is being removed. The following type codes are valid:

Type Code	Device
3179	3179 Display station
3180	3180 Display Station
3196	3196 Display Station
3197	3197 Display Station
3277	3277 Display Station
3278	3278 Display Station
3279	3279 Display Station
3476	3476 Display Station
3477	3477 Display Station
3486	3486 Display Station
3487	3487 Display Station
5251	5251 Display Station
5291	5291 Display Station
5292	5292 Color Display Station
5555	5555 Display Station (on systems supporting DBCS (double-byte character set))

***ALL:** The work station entry for all valid work station types is removed.

***NONASCII:** The work station entry for all valid work stations that use 5250 data stream is removed.

***ASCII:** The work station entries for all work stations that use ASCII data streams are added.

***CONS:** This value overrides a device type entry that specifies the same device type as the device being used as the console.

work-station-type: Specify the work station device type whose work station entry is being removed.

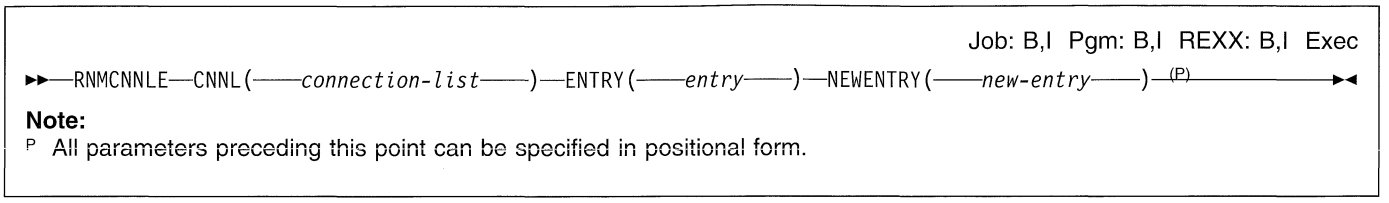
The WRKSTN parameter and the WRKSTNTYPE parameter are mutually exclusive.

Example

```
RMVWSE  SBSD(LIB2/CHARLES)  WRKSTN(B53)
```

This command removes the work station entry for work station B53 from the subsystem description named CHARLES in library LIB2.

RNMCNNLE (Rename Connection List Entry) Command



Purpose

The Rename Connection List Entry (RNMCNNLE) command renames an entry in the specified connection list.

Required Parameters

CNNL
Specifies the name of the connection list.

ENTRY
Specifies the name of the entry to be renamed.

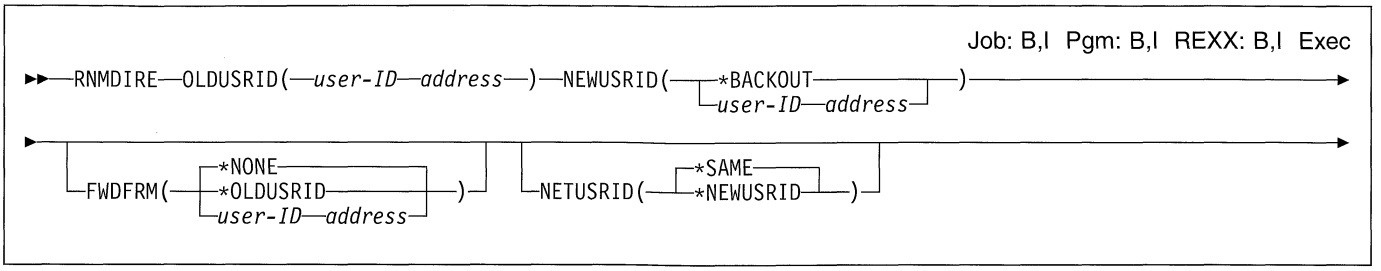
NEWENTRY
Specifies the new name for the entry. Each entry in the connection list must have a unique name.

Example

```
RNMCNNLE   CNNL(CHICAGO)
  ENTRY(CORPORATE)
  NEWENTRY(SERVICE)
```

This command renames entry CORPORATE in the connection list named CHICAGO. The new name will be SERVICE, but all other information for the entry remains the same.

RNMDIRE (Rename Directory Entry) Command



Purpose

The Rename Directory Entry (RNMDIRE) command renames a local or remote user identifier (ID) and user address to a new user ID and user address. A rename operation is not allowed for generic (*ANY) user IDs or default directory entries (QSYS, QDFTOWN, QLPAUTO, QLPINSTL). This command renames all occurrences of the specified user ID and address in all IBM-supplied files.

It is recommended that this job be scheduled during low-use periods using the Submit Job (SBMJOB) command. Rename is a long-running operation that does not allow OfficeVision/400 or calendar activity while it is running.

Only one rename operation can be run on the system at one time. If the rename is submitted to batch, the job waits for an active rename to complete.

Restrictions:

1. You must have security administrator (*SECADM) or all object (*ALLOBJ) authority to rename the user ID and user address.
2. The following must be ended before an entry can be renamed:
   - OfficeVision/400
   - PC Support
   - Hierarchical file support
   - QSNADS subsystem
   - TCP/IP electronic mail job (QTMSMTP)
   - Automatic cleanup through Operational Assistant*

   If any one of these are in use during the rename request, the request fails and an error message is sent.

Required Parameters

OLDUSRID

Specifies the user ID and address of the directory entry being renamed. Both elements must be specified. If lowercase characters are specified, the system stores them as uppercase characters. More information about specifying the user ID and address is in the *Distribution Services Network Guide*.

Element 1: User ID

user-ID: Specify the current user ID for the directory entry. A maximum of 8 characters can be specified. If this value is specified, an address must be specified on Element 2.

Element 2: Address

address: Specify the current address for the directory entry. A maximum of 8 characters can be specified. If this value is specified, a user ID must be specified on Element 1.

NEWUSRID

Specifies the user ID and address to which the old user ID and address is being renamed. Both elements must be specified but only one element needs to be different from the user ID and address specified on the OLDUSRID parameter.

The new user ID and address specified cannot be an existing user ID and address or exist as a forward-from value in the directory.

If the entry being renamed is in error from a previous rename request, you can continue with the rename operation or back out the changes and reset the files to the original values. To back out the changes, specify *BACKOUT on this parameter. To continue with the rename operation, do not change the value of this parameter (if the value is changed this is an error).

If lowercase characters are specified, the system stores them as uppercase characters.

***BACKOUT:** Back out of the rename directory entry operation. This value is only allowed on a directory entry that is in error as the result of a previous rename. This value sets the user ID and address in all IBM-supplied files changed by a previous rename request to the values specified on the OLDUSRID parameter.

Element 1: User ID

user-ID: Specify the new user ID for the directory entry. A maximum of 8 characters can be specified. If this value is specified, an address must be specified on Element 2.

Element 2: Address

address: Specify the new address for the directory entry. A maximum of 8 characters can be specified. If this value is specified, a user ID must be specified on Element 1.

Note: Changing the address element does not change the system name of the directory entry. If you want distributions for the user forwarded to a system other than what is specified by the directory entry, you must change the system name for the directory entry using the Change Directory Entry (CHGDIRE) command.

Optional Parameters

FWDFRM

Specifies whether distributions are automatically forwarded from the old user ID and address or a specified user ID and address. This value is valid only for local users.

***NONE:** Distributions are not forwarded.

***OLDUSRID:** All distributions are forwarded from the old user ID and address.

Element 1: User ID

user-ID: Specify the user ID from which distributions are to be forwarded. A maximum of 8 characters can be specified. If this value is specified, an address must be specified on Element 2.

Element 2: Address

address: Specify the address from which distributions are to be forwarded. A maximum of 8 characters can be specified. If this value is specified, a user ID must be specified on Element 1.

NETUSRID

Specifies whether the current network user ID and address are renamed to the new user ID and address. The network user ID is used in shadowing to uniquely identify a user in the network. The default is the user ID and address. If you are using directory shadowing with the user ID and address as the unique value in the network, you can also change this value to the new user ID and address specified on the NEWUSRID parameter.

***SAME:** The value does not change.

***NEWUSRID:** The network user ID and address are changed to the new user ID and address.

Examples

Example 1: Renaming a User ID

```
RNMDIRE  OLDUSRID(HURST PAYROLL)
  NEWUSERID(HURST NEWYORK)  FWDFRM(*OLDUSRID)
```

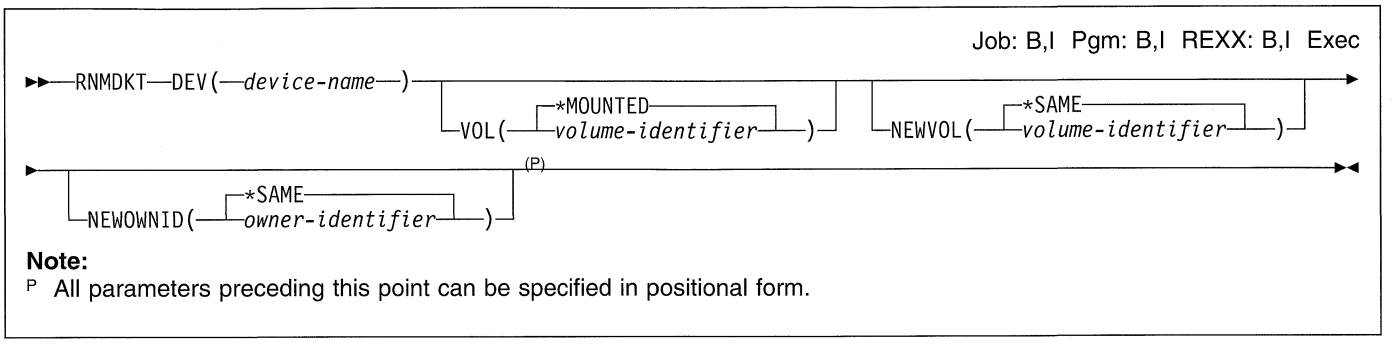
This command renames the current user ID HURST PAYROLL to the new user ID HURST NEWYORK. Distributions sent to the old user ID and address are forwarded.

Example 2: Renaming a User ID and Network User ID

```
RNMDIRE  OLDUSRID(HURST PAYROLL)
  NEWUSERID(HURST NEWYORK)  FWDFRM(*OLDUSRID)
  NETUSRID(*NEWUSRID)
```

This command renames the current user ID HURST PAYROLL and the current network user ID to the new user ID HURST NEWYORK. Distributions sent to the old user ID and address are forwarded.

RNMDKT (Rename Diskette) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Rename Diskette (RNMDKT) command changes the name of a diskette or changes the name (or identifier) of its owner. This command can be used to change the contents of the volume identifier field, or the owner identifier field, or both fields of a single diskette.

Note: When processing a diskette with labels that are not IBM standard labels, unpredictable results may occur. To initialize the diskette, enter the Initialize Diskette (INZDKT) command with CHECK(*NO) specified.

Required Parameters

DEV

Specifies the name of the diskette device containing the diskette being renamed.

Optional Parameters

VOL

Specifies one or more volume identifiers used by the file. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The volume currently placed in the device is used.

volume-identifier: Specify the volume identifier that is being compared with the diskette label volume identifier field on the diskette that is being renamed. Specify up to 6 characters; any combination of letters and numbers can be used.

If the volume identifiers do not match, a message is sent to the system operator. The operator can either insert the correct diskette and try again, or stop the command.

NEWVOL

Specifies, if the diskette is being renamed, the new volume identifier of the diskette.

***SAME:** The value does not change.

volume-identifier: Specify up to 6 characters for the volume identifier of the diskettes being renamed; any combination of letters and numbers can be used.

NEWOWNID

Specifies the owner identification being written in the volume label. The owner identification contains up to 14 characters (uppercase letters and/or numbers in any combination) and is left-justified and padded with blanks on the right if fewer than 14 characters are supplied.

***SAME:** The value does not change.

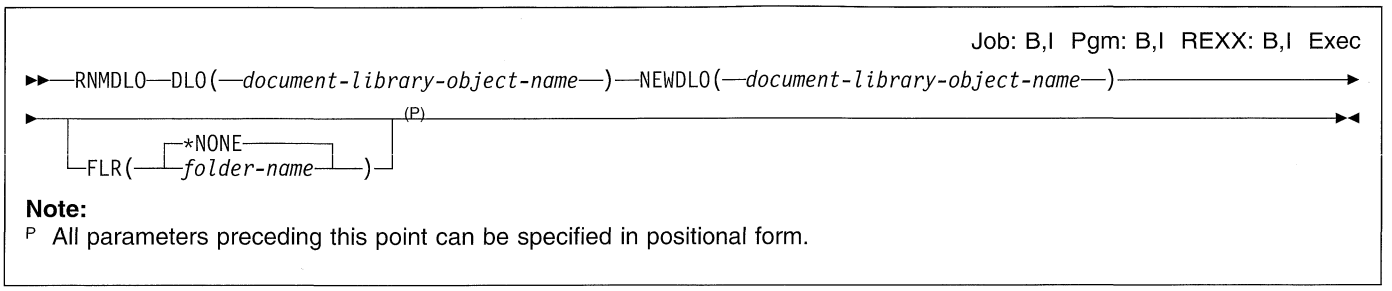
owner-identifier: Specify no more than 14 uppercase letters and numbers that identify the owner of the diskette. No lowercase letters, embedded blanks, or special characters can be included, even if they are enclosed in apostrophes. If less than 14 characters are used, the field is left-justified and padded on the right with blanks.

Example

```
RNMDKT  DEV(DKT1)  VOL(MASTER)  NEWVOL(BACKUP)
```

This command changes the name of the diskette in device DKT1, if its name is MASTER, to BACKUP. The owner identification is unchanged (NEWOWNID assumed to be *SAME).

RNMDLO (Rename Document Library Object) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Rename Document Library Object (RNMDLO) command changes the name of a document or folder. If the document or folder being renamed is in use when the command is used, the command is not run and a message is sent to the user who entered the command.

Restriction: The user must have *ALL authority to the document or folder being renamed, and must have *CHANGE authority to the folder that contains it. While using this command, the user may encounter an error message indicating that internal objects are locked. This means that another user is using the document library functions which cannot run at the same time as the RNMDLO command; therefore, retry this command later.

Required Parameters

DLO

Specifies the name of the document or folder that is being renamed.

NEWDLO

Specifies the new name of the document or folder.

Optional Parameters

FLR

Specifies the name of the folder that contains the document.

***NONE:** The folder being renamed is not located in a folder. *NONE cannot be specified if the object being renamed is a document.

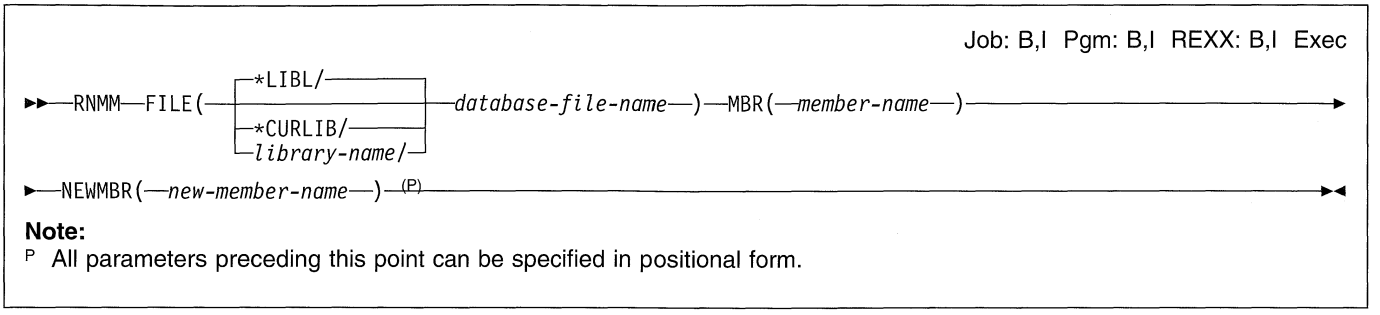
folder-name: Specify the name of the folder that contains the document or folder that is being renamed.

Example

```
RNMDLO  DLO(A)  NEWDLO(B)  FLR(FLR1)
```

This command changes the name of document or folder A located in folder FLR1, to B.

RNMM (Rename Member) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Rename Member (RNMM) command changes the name of a specified file member. The member cannot be renamed while it is in use; other users can read and change records of other members in the file that contains the member being renamed. A member that is opened in the same job cannot be renamed.

Restriction: To rename a file member, the user must have *ALL authority to the object being renamed and to the library in which the object is located.

Required Parameters

FILE

Specifies the qualified name of the database file (physical or logical) that contains the member being renamed.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the physical or logical database that contains the member being renamed.

MBR

Specifies the name of the physical or logical file member that is renamed. Specify the name of the member.

NEWMBR

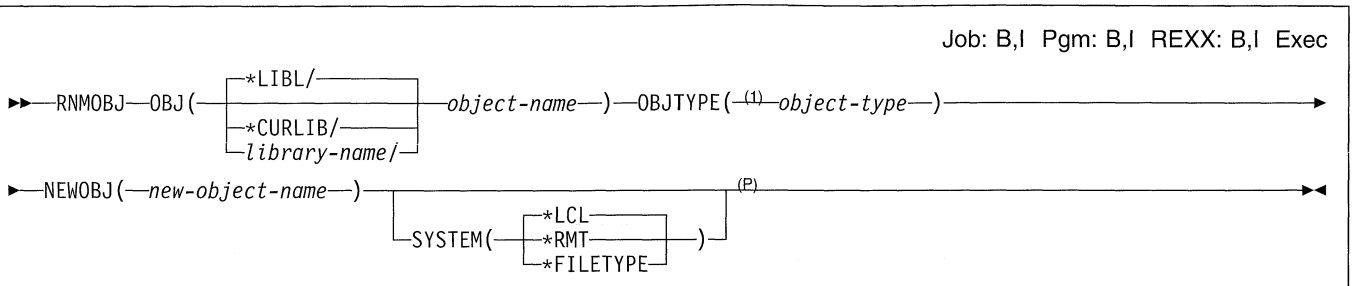
Specifies the new name of the member being renamed. The member remains in the same file. Specify the name of the new member. The new name must be unique in the file.

Example

```
RNMM  FILE(ELEMENT)  MBR(LEAD)  NEWMBR(GOLD)
```

This command renames member LEAD in file ELEMENT as GOLD. The library list (*LIBL) is used to find the file.

RNMOBJ (Rename Object) Command



Notes:

1 A list of the valid OS/400 object types for this command is in Appendix A.

P All parameters preceding this point can be specified in positional form.

Purpose

The Rename Object (RNMOBJ) command changes the name of an object in a library. The new name specified for the object must be unique in the library for the object type. If the object being renamed is in use when the command is entered, the command is not run and a message is sent to the user who enters the command. If a library is on an active user's library list when the library is renamed, a Display Library List (DSPLIBL) command reflects the new name. Renaming a library can cause programming errors. Therefore, it is not recommended.

Restrictions:

- The user must have object management authority for the object that is being renamed and have update authority for the library in which the object is located.
- A PL/I program cannot be renamed after it has been created.
- Configuration objects including controller descriptions, line descriptions, device descriptions, and network interface descriptions must all be varied off in order to rename them.
- The following objects cannot be renamed:
  - User profiles
  - Edit descriptions
  - Journals
  - Journal receivers
  - The job's temporary library (QTEMP)
  - The system library (QSYS)
  - The system operator message queue (QSYSOPR)
  - All work station user message queues
  - The system log (QHST)
  - The configuration objects (QCTL and QCONSOLE)
  - The configuration lists (QAPPNRMT, QAPPNLCL, QASYNCLOC, QRTLPASTHR)
  - The Electronic Customer Support configuration objects (QESLIN, QESPAP, QESCTL. QTILINE, QTICTL, QTIDA, QTIDA2, QIADSP, QIAPRT, QQAHOST)
- When renaming objects of type *CSI, *GSS, *FNTRSC, *FORMDF, *OVL, *PAGDFN, and *PAGSEG, the new name for the object cannot exceed 8 characters in length.

Notes:

1. References made to the following items may need to be updated by the user after a rename of a configuration object:
   - Connection lists
   - Work station entries
   - Communication entries
   - Display files
   - Printer files
   - Tape files
   - Diskette files
   - ICF files
   - User profiles
   - Job descriptions
   - System objects
   - CL programs
   - QPRTDEV system value
   - Display descriptions referencing this as an auxiliary printer
   - Communication side information (CSI) objects
   - Distributed data management files (APPC device name)
   - Integrated services digital network (ISDN) controller descriptions that refer to a renamed connection list (CNNL)
   - ISDN line descriptions that refer to a renamed CNNL
   - Other configuration objects. For example, lines, controllers, and other devices that refer to the renamed configuration objects.
2. References made to the renamed object by the following items are automatically changed by the system after a rename operation. The reference changes reflect the changes made to the renamed configuration objects.
   - QCONSOLE system values
   - message queues associated with display devices

- System/36 environment device tables
- output queues associated with the old printer device
- local work station controllers associated with a twinaxial data link control (TDLC) line
- TDLC lines associated with the local or remote work station controller

Required Parameters

OBJ

Specifies the current qualified name of the object to be renamed. If no library qualifier is given, *LIBL is used to find the object. The object name should be qualified to ensure that the right object is renamed.

The name of the object can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

object-name: Specify the current name of the object to be renamed.

OBJTYPE

Specifies the type of the object to be renamed. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: Dependent logical files follow the renamed physical file.

NEWOBJ

Specifies the new name of the object being renamed. The object remains in the same library. Enter the new name that identifies the object to the system.

Optional Parameters

SYSTEM

Specifies whether the file is on the local system (*LCL), the remote system or systems (*RMT), or on either the local or remote systems (*FILETYPE).

***LCL:** The rename is for a file on the local system.

***RMT:** The remote file referred to by the source distributed data management (DDM) file is renamed.

Note: If the user wants to rename a remote file, two DDM files must be used. The existing DDM file goes in the OBJ value and the new DDM file goes in the NEWOBJ value. The new DDM file must be in the same library as the existing DDM file. When the remote rename occurs, the remote file name in the existing DDM file is given to the new DDM file.

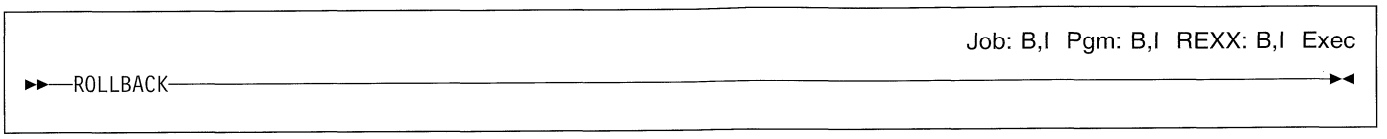
***FILETYPE:** If the OBJ name is a DDM file, the renaming is remote, but if the OBJ name is not a DDM file, the renaming is local.

Example

```
RNMOBJ  OBJ(PAYROLL/FILEX)  OBJTYPE(*FILE)
  NEWOBJ(MSTR)
```

The library named PAYROLL is searched for the file named FILEX. If the file is found, and the user has object operational authority for FILEX and update authority for the PAYROLL library, FILEX is renamed MSTR.

ROLLBACK (Rollback) Command



Purpose

The Rollback (ROLLBACK) command is used to restart the current transaction and reestablish the last commitment boundary as the current commitment boundary for the commitment definition associated with the program issuing the command.

When the ROLLBACK command is issued:

- Changes made to database files and other commitment resources under commitment control for the commitment definition since the last commitment boundary was established are rolled back. Updates, additions, or deletions made to the database file's data since that commitment boundary are rolled back or removed, and the original entries are put back in the files. Records that were added to the files remain as deleted records. The files are repositioned to the last commitment boundary. Changes made to other commitment resources are rolled back as well.
- All record locks held for files opened under commitment control for the commitment definition are released.
- Locks on object level commitment control resources, acquired when the resources were created or changed during the transaction, are released.

More information on the use of commitment control is in the *Advanced Backup and Recovery Guide*.

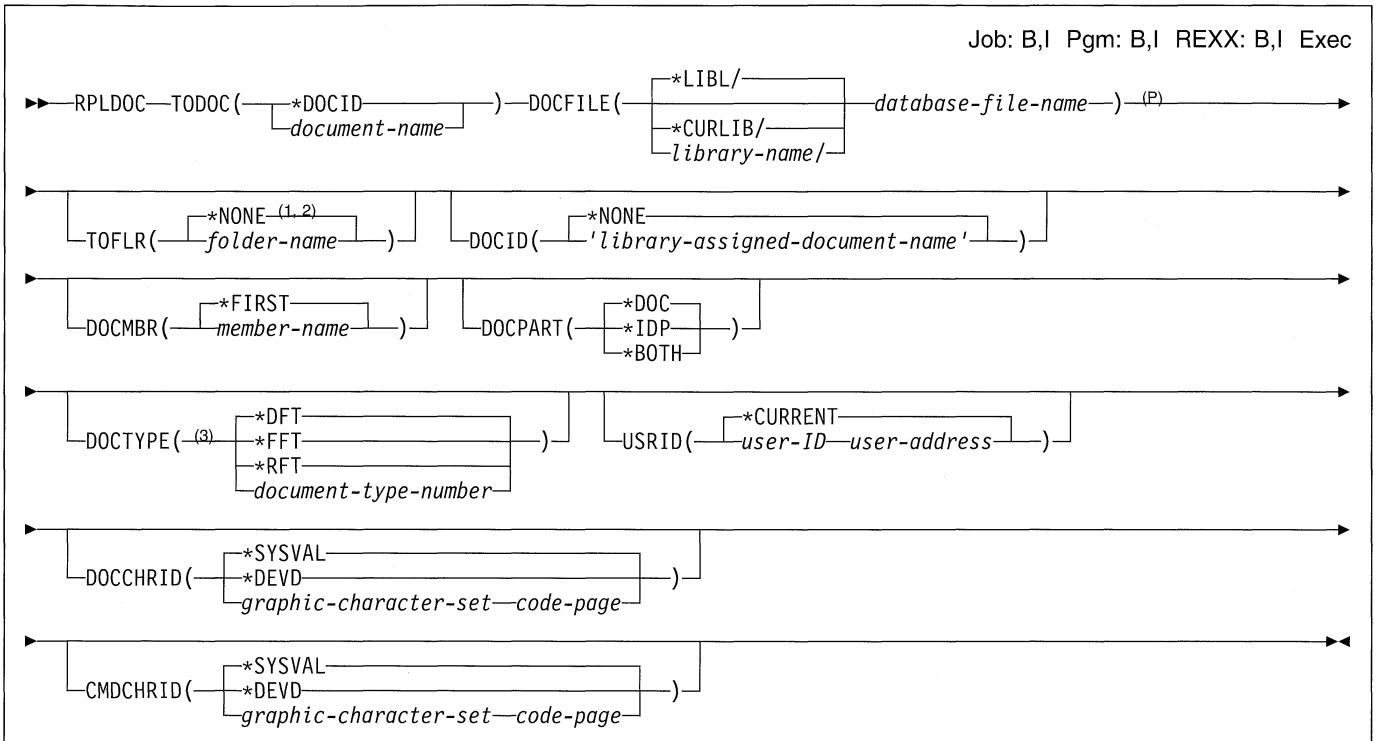
There are no parameters for this command.

Example

```
ROLLBACK
```

This command reestablishes the last commitment boundary (the point at which a Commit (COMMIT) command or Rollback (ROLLBACK) command was last issued) for the commitment definition associated with the program issuing the command.

RPLDOC (Replace Document) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 This value cannot be specified when a document name is specified on the TODOC parameter.

2 This value must be specified when a document is specified on the DOCID parameter.

3 This parameter is ignored if only the IDP is being replaced.

Purpose

The Replace Document (RPLDOC) command is used to replace the document content and the interchange document profile (IDP) of a document that exists in the document library.

Restrictions:

1. The document must be checked out with the Retrieve Document (RTVDOC) command by the user specified on the USRID parameter before using this command.
2. The user must have at least *CHANGE authority for the document, *ALLOBJ authority, or be working on behalf of a user who is authorized for the document.
3. Authority to work on behalf of another user is granted with the Grant User Permission (GRTUSRPMN) command.

Required Parameters

TODOC

Specifies the name of the document where the data is being placed, or the DOCID parameter is used to specify the library assigned document name where the data is being replaced.

***DOCID:** The document being replaced is identified by the library-assigned document name specified on the DOCID parameter.

document-name: Specify the name of the document that is replaced.

DOCFILE

Specifies the qualified name of the database file that contains the document data being filed. The database file can be user-defined or it can be the output file specified on the Receive Distribution (RCVDST) or Retrieve Document (RTVDOC) command. Database files that are used to replace document data are used as is with any changes. If an output file is specified, only the document data and IDP record format are read from the output file and the prefix is removed from the document data. More information on defining the format of database files is in the *Database Guide*.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file.

Optional Parameters

TOFLR

Specifies the name of the folder that contains the document being replaced.

***NONE:** No folder name is specified.

folder-name: Specify the name of the folder that contains the document being replaced. A folder name can consist of a series of folder names if the document being replaced is located in a folder that is contained in another folder. Up to 63 characters can be specified.

DOCID

Specifies the library-assigned name of the document being replaced. This is the name assigned to the document by the system when it is created. The library-assigned document name can be determined by using the Query Document Library (QRYDOCLIB) command. The library-assigned name is also returned in a completion message when using the File Document (FILDOC) command.

Library-assigned document names are 24 characters in length with the following format:
YYYYMMDDHHMNSSHSSNSNSNSN where

YYYY = year
MM = month
DD = day
HH = hour
MN = minute
SS = second
HS = hundredths of a second
SNSNSNSN = system name

***NONE:** No library-assigned document name is required when the document is identified by the TODOC parameter.

'library-assigned-document-name': Specify the library-assigned document name to be replaced. More information on library-assigned document names is in the *Communications: Distribution Services Network Guide*.

Note: When the user is replacing a document and a cancel request is entered, the document may or may not be replaced.

DOCMBR

Specifies the document database file member that contains the data and IDP to replace the current data and IDP in the document.

***FIRST:** The first member (in order of creation) in the database file contains the replacement data.

member-name: Specify the name of the database file that contains the replacement data.

DOCPART

Specifies the part of the document to be replaced.

Note: If you specify DOCPART(*BOTH) and the replacement of one fails, neither is replaced and the document remains checked out.

***DOC:** Replace the document content only. If no document content records exist in the specified file, the document is replaced but a message is returned to alert the user that no document content was replaced.

***IPD:** Replace only the IDP of the document. If this value is specified, IDP records (record code 500) must exist in the specified file or the command fails.

***BOTH:** Replace both the document content and the IDP of the document. If this value is specified, IDP records (record code 500) must exist in the specified file or the command fails.

DOCTYPE

Specifies the type of replacement document. This identifier is used by the system to determine whether it can handle the data stream correctly.

The system code (SYSCOD on the File Document (FILDOC) command) cannot be changed when the document contents are replaced.

***DFT:** The system creates the appropriate document type identifier depending on where the data comes from.

If the document file is an output file created by the RCVDST or RTVDOC command, then the document type is taken from this file. If the file does not contain a document type or if the value in this file is document type 0, then an error message is sent. If the document file is a user-defined file, then the document type is 223 (data file).

Document Type	Description
2	Final form text document (FFTDCA)
3	5520 revisable form text document (RFT5520)
4	Word processing EBCDIC (EBCDIC)
5	Word processing information file (WRDPRCINFF)
6	Image-data subset document (IMAGE)
7	3730 text data stream (TEXT3730)
8	DIA document library descriptor document (DOCDESCDOC)
9	3732 display document data stream (DATAST3732)
10	DIA display document data stream (DOCUNITCTL)
11	Revisable form text document (RFTDCA)

12	1403 compatible data stream with variable length, unblocked records (PRT1403)
13	Digitized ADS audio (AUDIO)
14	IBM PC data file (PCFILE)
15	Hard copy of document (HARDCOPY)
223	Data file
32753	List of documents created from search (DOCLIST)
32756	Version 2 DIA document, library document, descriptor document
32768	AS/400 system revisable-form document (RFTAS400)
32769	AS/400 system final-form document (FFTAS400)
32770	IBM DW4 revisable-form document (RFTDW4)
32912	ASCII data (ASCIIDATA)
65262	IBM S/38 revisable document (RFTS38)

***FFT:** The document is Final Form Text. This type of document is intended to be shown and printed and not edited by the receiver.

***RFT:** The document is Revisable Form Text. This type of document can be shown, printed, and edited by the receiver.

document-type-number: Specify a number from 2 through 65535. The numbers from 2 through 32767 are controlled by registering them with the IBM Document Interchange Architecture and are used for IBM-defined document types. The numbers 32768 through 65535 are not registered with IBM and can be used for non-IBM defined document types.

USRID

Specifies the user ID and address of the user for whom this request is made. The current user of this command must have the authority to work on behalf of the named user ID. Authority to work on behalf of another user is given with the Grant User Permission (GRTUSRPMN) command.

***CURRENT:** The user profile under which the current job is running is used.

Element 1: User ID

user-ID: Specify the user ID of the user whose the document is replaced.

Element 2: User Address

user-address: Specify the user address of the user for whom the document is replaced.

DOCCHRID

Specifies the character identifier (graphic character set and code page) for the data being filed. The character identifier is related to the display station that was used to create the data being filed. More information about CHRID processing is in the *Database Guide*.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set used to create the data being filed.

Element 2: Code Page

code-page: Specify the code page value used to create the command parameters. Valid values range from 1 through 999.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the *Guide to Programming Displays*.

Note: This value translates the USRID parameter to the character set and code page of '930 500'. The *Communications: Distribution Services Network Guide* contains the character set and code page table for '930 500'.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values used to create the command parameter. Valid values range from 1 through 999.

Element 2: Code Page

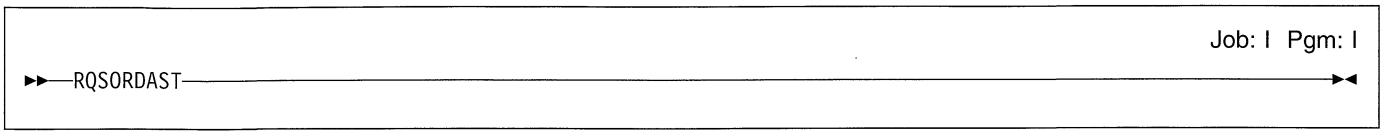
code-page: Specify the code page value used to create the command parameters. Valid values range from 1 through 999.

Example

```
RPLDOC  TODOC(*DOCID)  DOCFILE(*LIBL/MYFILE)
  DOCPART(*BOTH)  DOCID('1987060710102053SYSTEM1')
  DOCTYPE(*FFT)
```

This command replaces the document data and IDP with data in the file MYFILE. The data is placed in the document identified by the document identifier '1987060710102053SYSTEM1'. The document type is changed to Final Form Text.

RQSORDAST (Request Order Assistance) Command



Purpose

The Request Order Assistance (RQSORDAST) command sends a request to IBM for order assistance. You can request assistance in ordering services and products including:

- Software upgrades
- Hardware upgrades
- AS/400 publications
- Service offerings
- General help (for example, network planning)

When the RQSORDAST command is successfully processed, a file containing the order information is created and sent along with the order assistance request. This file contains:

- Hardware configuration information (vital product data (VPD) and topology data)
- Software configuration information (installed IBM program products)

The order information file is sent with all requests.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. You must have *ALLOBJ authority or be signed on as QSYSOPR or QSRV to use the command.

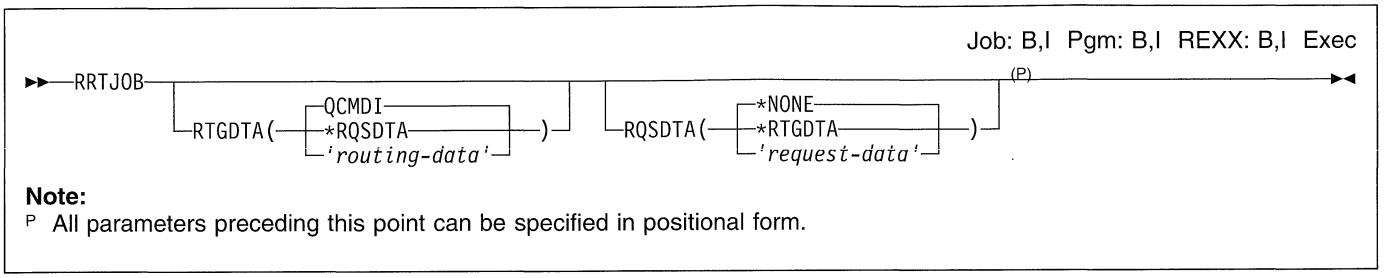
There are no parameters for this command.

Example

```
RQSORDAST
```

This command displays the Request Order Assistance entry panel.

RRTJOB (Reroute Job) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Reroute Job (RRTJOB) command starts a new routing step for a job in the current subsystem. For example, the job may need rerouting so that it runs under a different class or in a different storage pool. The rerouting requires changes in the routing data for the job, and calls a different program used with the new routing step.

When this command is used, objects allocated in the previous routing step are deallocated and open files are closed. If the objects or files are needed in the new routing step, they must be allocated or opened again.

Note: Running of this command in a batch job causes loss of spooled inline files because they cannot be accessed in the new routing step. Also, if the RRTJOB command is run while the system is ending (by running of a End System (ENDSBS) command, End System (ENDSYS) command, or the Power Down System (PWRDWNSYS) command), a new routing step is not started and the job is ended.

Restriction: The job must not be a group job.

Optional Parameters

RTGDTA

Specifies the routing data used with this job description to start jobs. The routing data is used to determine the routing entry (in the subsystem description) that identifies the program in which the job runs.

QCMDI: This routing data matches a routing entry in the IBM-supplied subsystem description (QINTER), which indicates a routing step that is processed by the IBM-supplied control language processor, QCMD, in the QSYS library.

***RQSDTA:** The first 80 characters of the request data specified in the RQSDTA parameter of this command is also used as the routing data for the next routing step.

'routing-data': Specify the character string used as the routing data for starting the next routing step. Up to 80 characters can be entered, enclosed in apostrophes if necessary.

RQSDTA

Specifies the request data that is added on the end of the job's message queue for use by the new routing step. For example, if RTGDTA (QCMDB) is specified, the IBM-supplied batch subsystem QBATCH is being used, and a CL command is supplied, it becomes a message that is read by the control language processor, QCMD (if the submitted job is routed to QCMD).

***NONE:** Request data is not placed in the job's message queue.

***RTGDTA:** Routing data specified in the RTGDTA parameter is also placed at the end of the job's message queue.

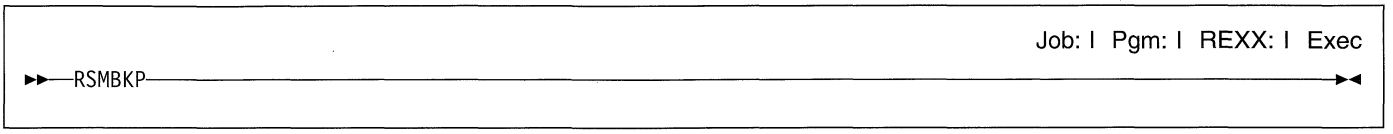
'request-data': Specify the character string placed at the end of the job's message queue for use by the new routing step or some subsequent routing step in the job. Up to 256 characters can be entered, enclosed in apostrophes if necessary. When a CL command is entered, it must be enclosed in single apostrophes, and where apostrophes would normally be used *inside* the command, double apostrophes must be used.

Example

```
RRTJOB  RTGDTA(INQUIRY)
```

This command reroutes the job in which the command is issued by starting a new routing step with the routing data INQUIRY. The job remains in the same subsystem.

RSMBKP (Resume Breakpoint) Command



Purpose

The Resume Breakpoint (RSMBKP) command causes a program to continue processing after it has been stopped at a breakpoint. The program that continues is the one that most recently stopped at a breakpoint. When more than one program in the job is stopped at a breakpoint, the End Request (ENDRQS) command can be used to return to the Command Entry Display for a previous program call that is also stopped at a breakpoint.

If you are servicing another job, and it has not ended, this command resumes that job from the breakpoint. It also returns the servicing job to the point immediately preceding where the breakpoint display was shown.

Restriction: This command is valid only for programs in the debug mode and only when the program is stopped at a user-defined breakpoint. That is, this command is not valid at a breakpoint caused by an unmonitored message. To start the debug mode, use the STRDBG (Start Debug) command.

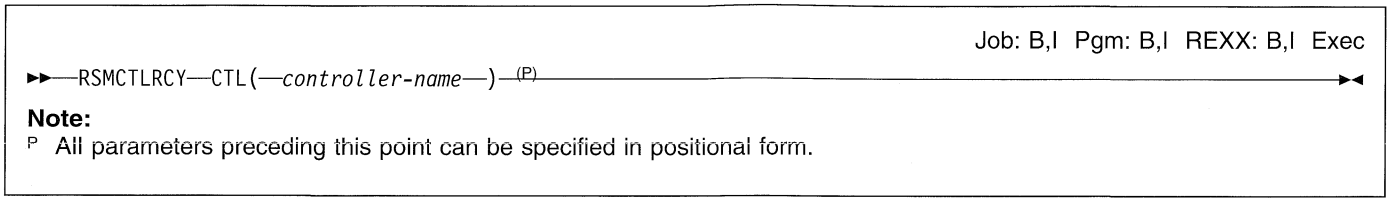
There are no parameters for this command.

Example

```
RSMBKP
```

Assuming that the program having control is stopped at a breakpoint, this command causes the program to continue processing, starting from the breakpoint location.

RSMCTLRCY (Resume Controller Recovery) Command



Purpose

The Resume Controller Recovery (RSMCTLRCY) command resumes error recovery procedures for a specific controller. The error recovery procedures can be ended by the End Controller Recovery (ENDCTLRCY) command or by responding to a failure-related inquiry message with a cancel option.

The RSMCTLRCY command allows the user to resume automatic error recovery procedures after they have been stopped, and to reactivate a controller after it has been canceled (if the C response was entered to the inquiry message associated with a controller failure). When the controller is canceled with the C response, all jobs are ended; once the controller is repaired and the RSMCTLRCY command is entered, jobs are allowed to start using the controller again.

Restriction: To use this command, the user must have object operational authority for the controller.

Required Parameters

CTL

Specifies the controller whose recovery is started again. Valid types of controllers are:

CTL Value	Controller
5251	Display station
*PU2	Physical unit (type 2); SDLCs for basic BSC and RJE
*BSC	BSC device (basic BSC and RJE)
*BSCT	BSC device (Multipoint tributary and 3270 Device Emulation)
*APPC	Advanced program-to-program communications
*WSC	Local work station
*WSCE	Local work station (extended)

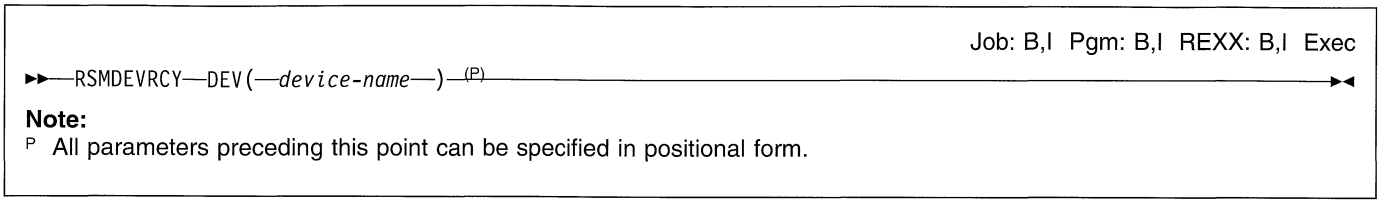
Specify the name of the controller (as specified in the controller description) for which error recovery procedures are to resume.

Example

```
RSMCTLRCY  CTL(TROLL3)
```

This command resumes error recovery procedures for the controller TROLL3.

RSMDEVRCY (Resume Device Recovery) Command



Purpose

The Resume Device Recovery (RSMDEVRCY) command resumes error recovery procedures for a specific device. The error recovery procedures are ended by the ENDDEVRCY command or by responding to a failure-related inquiry message with a cancel option.

The RSMDEVRCY command allows you to resume automatic error recovery procedures after they have been stopped, and to reactivate a device after it has been canceled (if you entered the C response to the inquiry message associated with a device failure). When the device is canceled with the C response, all jobs are ended; once the device is repaired and the RSMDEVRCY command is entered, jobs are allowed to start using the device again.

Restriction: To use this command, you must have object operational authority for the device.

Required Parameters

DEV

Specifies the device whose recovery is started again. Valid types of devices are:

DEV Type	Device
5219	Printer (work station)
5224	Printer (work station)
5225	Printer (work station)
5251	Display station
5252	Dual display station
5256	Printer (work station)
5291	Display station
5292	Display station
*PLU1	Physical unit (type 1)
*BSC	BSC device
*BSCT	BSC multipoint tributary
*APPC	Advanced program-to-program communications

Specify the name of the device (as specified in the device description) for which error recovery procedures are to resume.

Example

```
RSMDEVRCY  DEV(WSPR03)
```

This command resumes error recovery procedures for the device WSPR03 to resume.

RSMLINRCY (Resume Line Recovery) Command

Job: B,I Pgm: B,I REXX: B,I Exec

Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Resume Line Recovery (RSMLINRCY) command resumes error recovery procedures for a specific line. The error recovery procedures are ended by the End Line Recovery (ENDLINRCY) command or by responding to a failure-related inquiry message with a cancel option.

The RSMLINRCY command allows the user to resume automatic error recovery procedures after they stop, and to reactivate a line after it is canceled (if the C response was entered to the inquiry message associated with a line failure). When the line is canceled with the C response, all jobs are ended; once the line is repaired and the RSMLINRCY command is entered, jobs are allowed to start using the line again.

Restriction: To use this command, the user must have object operational authority for the line.

Required Parameters

LINE

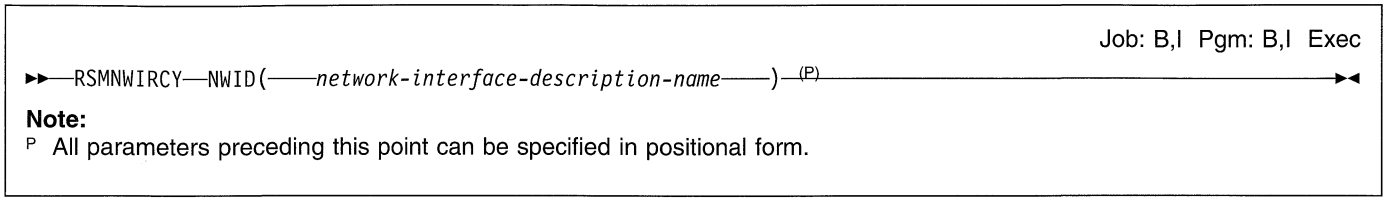
Specifies the line whose recovery is started again.

Example

```
RSMLINRCY  LINE(NYC2)
```

This command resumes error recovery procedures for the line NYC2 to resume.

RSMNWIRCY (Resume Network Interface Recovery) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Resume Network Interface Recovery (RSMNWIRCY) command is used to resume error recovery procedures for a network interface description. Error recovery procedures are ended by the End Network Interface Recovery (ENDNWIRCY) command or by responding to a failure-related inquiry message with a cancel option. This command is used to resume automatic error recovery procedures after they stop, and to reactivate a network interface description (and jobs using that description) after it is canceled.

Required Parameters

NWID

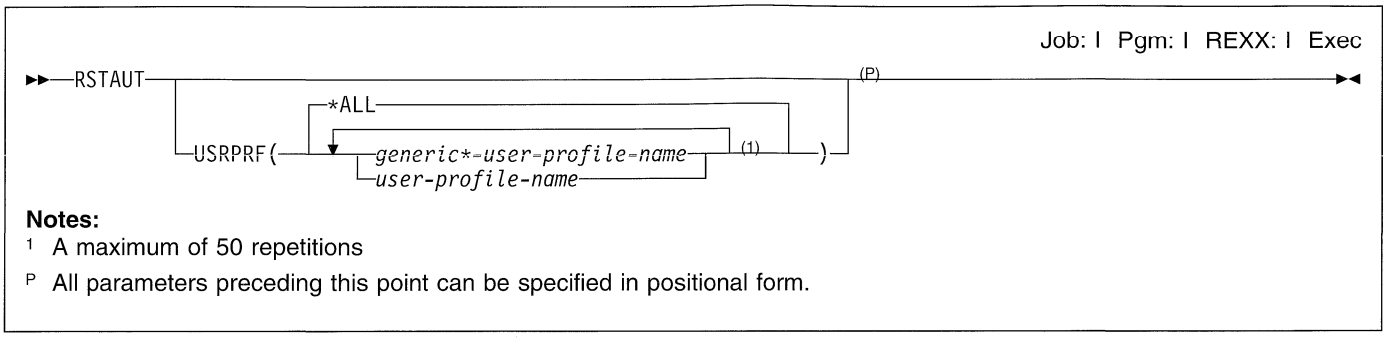
Specifies the name of the network interface description whose automatic error recovery is restarted.

Example

```
RSMNWIRCY   NWID(ISDNNET)
```

This command resumes error recovery procedures for the network interface description named ISDNNET.

RSTAUT (Restore Authority) Command



Purpose

The Restore Authority (RSTAUT) command restores the private authorities to user profiles. This command restores the same object use authority to specified objects in the user profile that each user profile had when all the profiles were saved by the Save System (SAVSYS) or Save Security Data (SAVSECDTA) commands. It allows existing authorities, given after the save, to remain. Authority is not restored to the user profiles until the profiles are first restored to the system by the Restore User Profiles (RSTUSRPRF) command and all the objects (for which authority is being given) are restored to the same libraries where they are saved. The objects can be restored with the Restore Library (RSTLIB) or Restore Object (RSTOBJ), Restore Document Library Object (RSTDLO), or RSTCFG (Restore Configuration) commands.

If the whole system is being restored, the following sequence must be followed. Using the RSTAUT command must be the last step in the sequence.

1. Restore the operating system. This is an alternative method to load the program. This restores the QSYS library and ensures that the IBM-supplied user profiles are there.
2. Restore all the saved user profiles to the system (*ALL is the default for the USRPRF parameter) by using the RSTUSRPRF command.
3. Restore all the configuration and system resource management (SRM) objects to the system by using the RSTCFG command.
4. Restore all the user libraries (including the other IBM-supplied libraries) by using the RSTLIB command. Unless saved with the *IBM or *ALLUSR parameter, *NONSYS must be specified on the SAVLIB parameter to restore all user libraries in one operation.
5. Restore all document library objects to the system by using the RSTDLO command.
6. Restore the object authority to user profiles by using the RSTAUT command.

Note: Steps 2 through 6 can be done more than once. For example, after the user profiles are restored (step 2), the user can restore only critical application libraries (step 3), followed by a restore of object authority (step 6). This supplies an operational system limited to using only the critical libraries. Later, the remaining user profiles can be restored, followed by the operations to restore the libraries and object authority.

If authorities for a user profile are restored using the RSTAUT command, the user profile must be restored again before other authorities for it can be restored.

If one user profile is being restored, the following sequence must be followed. Using the RSTAUT command must be the last step.

1. Restore the specified user profile to the system by using the RSTUSRPRF command.
2. Restore all the device configuration and SRM objects to the system by using the RSTCFG command.
3. Restore the specified user libraries to the system by using the RSTLIB command or the RSTOBJ command. If the user profile is being restored because the current profile on the system is damaged, then the needed libraries already exist on the system and restoring of the libraries is not necessary.
4. Restore all document library objects to the system using the RSTDLO command.
5. Restore the object authority to the user profile by using the RSTAUT command. The specified profile may have been restored using the RSTUSRPRF command.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. You must have *SAVSYS special authority to use this command.
3. All subsystems must be ended (ENDSYS or ENDSBS(*ALL)) before running this command.

Optional Parameters

USRPRF

Specifies the names of one or more user profiles whose private authorities are being restored. The specified

user profiles must first be restored using the RSTUSRPRF command.

***ALL:** Private authorities are being restored for all of the user profiles that are restored but do not have their private authorities restored. This includes user profiles that were restored using multiple RSTUSRPRF commands.

generic-user-profile-name:* Specify the generic name of the user profile. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk (*) substitutes for any valid characters. For more information on the use of generic functions, refer to "Rules for Specifying Names."

user-profile-name: Specify one or more names of specific user profiles. Both generic names and specific names can be specified in the same command.

Examples

Example 1

```
RSTAUT
```

This command restores to each user profile the authority to use each object that the profile had at the time when the system was saved. The user profiles and the libraries and their objects must be restored before the RSTAUT command is sent.

Example 2

```
RSTUSRPRF USRPRF(USER1 USER2 USER3 USER4)

RSTLIB SAVLIB(USERLIB)

RSTAUT USRPRF(USER1 USER2 USER3)
```

To each specified user profile that was successfully restored, this command restores the authority to use each object that the profile had at the time the system was saved. The user profiles and the libraries and their objects must be restored before the RSTAUT command is sent. Because USER4 was not specified in the RSTAUT command, its authorities are still available and may be restored at a later date.

Example 3

```
RSTUSRPRF USRPRF(*ALL)

RSTLIB SAVLIB(USERLIBA)

RSTLIB SAVLIB(USERLIBB)

RSTLIB SAVLIB(USERLIBC)

RSTAUT USRPRF(*ALL)
```

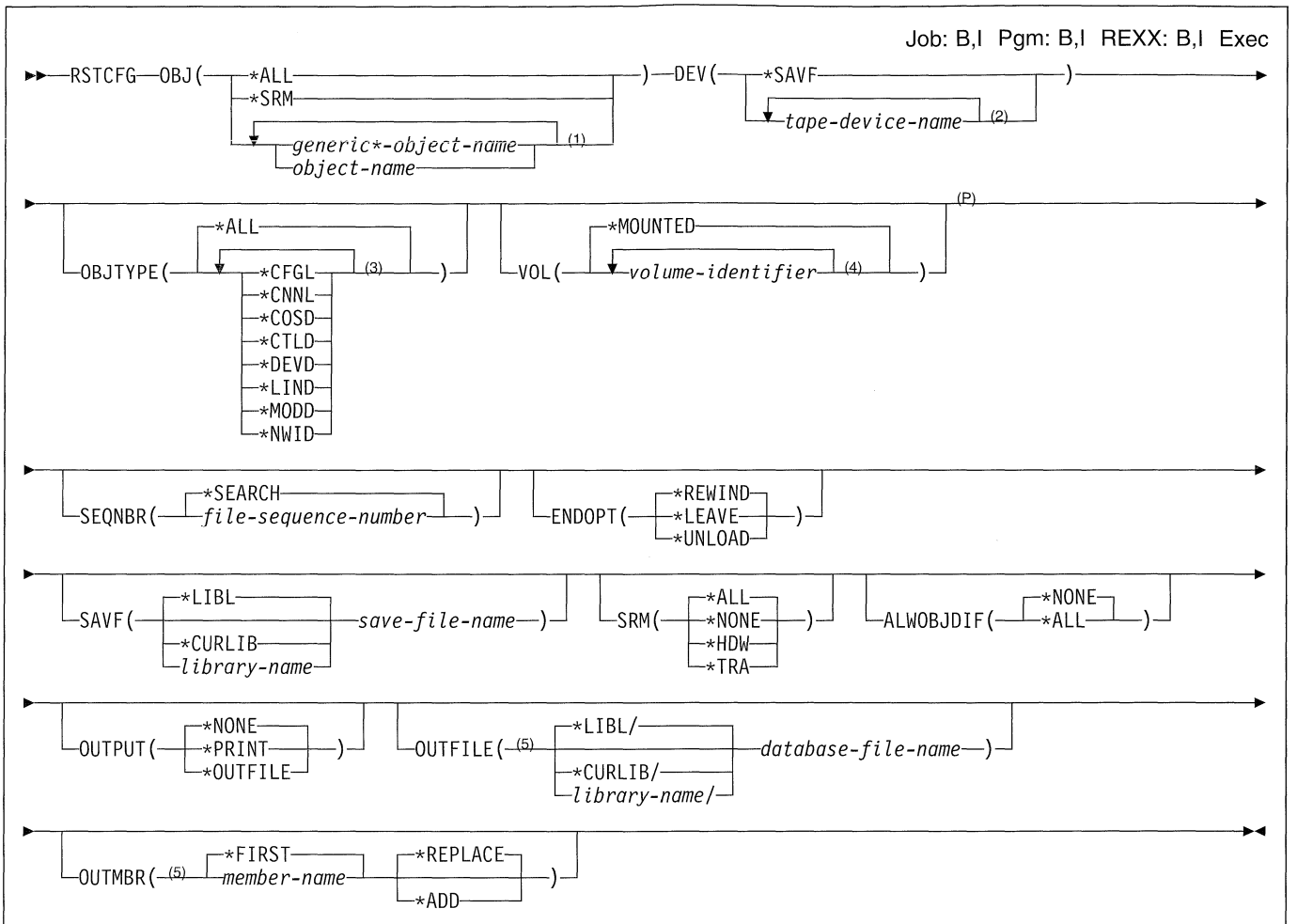
This command restores private authorities for all restored user profiles on the system. This includes authorities for all user profiles restored by the RSTUSRPRF command. Other user profiles on the system that did not have their authorities restored before these commands were specified are also restored by the RSTAUT(*ALL) command.

Example 4

```
RSTUSRPRF USRPRF(USER1 USER2)

RSTLIB SAVLIB(USERLIBA)

RSTUSRPRF USRPRF(USER1 USER3)

RSTLIB SAVLIB(USERLIBB)

RSTAUT USRPRF(*ALL)
```

This command restores private authorities for USER2 and USER3 and for the most recent version of USER1. Because the user profiles have the same name, the second RSTUSRPRF command overlays the first version of USER1.

RSTCFG (Restore Configuration) Command



Notes:

1 A maximum of 300 repetitions

2 A maximum of 4 repetitions

3 A maximum of 8 repetitions

4 A maximum of 75 repetitions

5 This parameter is valid only if OUTPUT(*OUTFILE) is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Restore Configuration (RSTCFG) command restores to the system a device configuration object that was saved to tape by the Save System (SAVSYS) or Save Configuration (SAVCFG) command. The types of objects that can be restored by this command are listed in the OBJTYPE parameter.

The user profile of the system default owner (QDFTOWN) becomes the default owner of any objects being restored on the system when the profile of the owner is not known to the system.

If an object already exists, in the library to which the object is restored, the public and private authorities of the existing object are kept. If the object does not exist in the library, all public authorities are restored, but any private authorities must be granted again.

Restrictions:

1. To use this command, the user must have *SAVSYS authority, or object existence authority for (or be the owner of) each object specified if the object already exists on the system.
2. This command is shipped with public *EXCLUDE authority.

3. The device configuration object must be varied off when it is being restored. To vary off a device configuration object, use the Vary Configuration (VRYCFG) command.
4. With the exception of overrides for the output listing file, this command ignores all file overrides currently in effect for the job.
5. System resource management (SRM) objects are not restored if the RSTCFG command is run using media that was created prior to V2R2M0.
6. If the RSTCFG command and the SAVSYS or SAVCFG commands are not run on the same system, the configuration objects may not match the physical hardware on the target system.
7. If you restore system resource management objects on a system other than the one on which the SAVSYS or SAVCFG command was saved, the system then treats the target system hardware as new and creates all new resource names, making the existing configuration descriptions useless. If this occurs, you need to restore the correct system resource management database from the most current SAVSYS or SAVCFG command for that command. If neither of these is available, you must change existing configuration descriptions to reflect the new resource names.

Required Parameters

OBJ

Specifies the names of one or more configuration objects being restored.

System resource management (SRM) objects cannot be restored individually or by specifying a generic name. To restore only SRM objects, specify *SRM on the OBJ parameter and a value on the SRM parameter.

***ALL:** All the device configuration objects are restored, depending on the values specified for the OBJTYPE parameter.

***SRM:** The device configuration objects are not restored, but system resource management (SRM) objects are restored based on the SRM parameter value.

generic-object-name:* Specify the generic name of the object. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix, for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

object-name: Specify one or more names of specific objects being restored. Both generic names and specific names can be specified in the same command. A maximum of 300 object names can be specified.

DEV

Specifies the name of the tape device used to restore the objects.

***SAVF:** The save file specified on the SAVF parameter is used for the restore operation.

tape-device-name: Specify the names of one or more tape devices used for the restore operation. If you are using more than one tape device (up to a maximum of four), specify the names of the devices in the order in which they are used.

The device name must already be known on the system by a device description. If using more than one tape drive (up to four), specify the names of the drives in the order they are used. If the objects are on more than one tape volume, the user may want to use more than one tape drive, so that one tape volume can be rewound and unloaded while another tape drive processes the next volume.

Optional Parameters

OBJTYPE

Specifies the types of OS/400 system objects that are restored.

***ALL:** All device configuration object types that are specified by name are restored. If *ALL is also specified on the OBJ parameter, all of the saved device configuration objects are restored.

object-type: Specify the value for each of the types of objects that are being restored. The following types can be specified:

***CFGL**	Configuration list
***CNNL**	Connection list
***COSD**	Class-of-service description
***CTLD**	Controller description
***DEVD**	Device description
***LIND**	Line description
***MODD**	Mode description
***NWID**	Network interface description

The object types shown are the ones saved in the device configuration media file by using the Save System (SAVSYS) or Save Configuration (SAVCFG) command.

Note: *SRMSPC are also saved, but not restored as an object type. Specify *SRM on the OBJ parameter to restore SRM data.

VOL

Specifies the volume identifiers of the tape volumes from which the objects are being restored. The volumes must be placed in the tape device in the same order as they were when the objects were saved. The volume that contains the beginning of the file to be restored should be placed in the tape unit. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: The version of the objects that is restored is the first version found in the specified location.

***MOUNTED:** The objects are restored from the volumes placed in the device specified on the DEV parameter.

volume-identifier: Specify the identifiers of one or more volumes in the order in which they are put on the device and used. Each volume identifier contains up to 6 alphanumeric characters. A blank is used as a separator character when listing multiple identifiers.

SEQNBR

Specifies which sequence number is used for the restore operation.

***SEARCH:** The volume placed in a device is searched for a data file containing the saved device configuration objects. When a match is found, the configuration objects are restored. If the last operation on the device specifies ENDOPT(*LEAVE), the tape remains positioned at the location where the last operation ended. The file search starts with the first data file beyond the current tape position. If ENDOPT(*LEAVE) was not used for the last operation, or if the tape was manually rewound after an ENDOPT(*LEAVE) operation, the search starts with the first data file on the volume.

file-sequence-number: Specify the sequence number of the file that is used.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

SAVF

Specifies the qualified name of the save file used to restore the save data.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the save file.

SRM

Specifies the type of system resource management (SRM) information to be restored. This parameter is valid only when *ALL or *SRM is specified on the OBJ parameter.

***ALL:** All SRM information is restored.

***NONE:** No SRM information is restored.

***HDW:** All hardware information is restored.

***TRA:** All token-ring adapter information is restored.

ALWOBJDIF

Specifies whether certain differences encountered during a restore operation are allowed. There are two differences that apply to the RSTCFG command:

- The owner of the object on the system is not the owner of the object from the save operation.
- The object is secured by an authorization list and is being restored to a system other than the one on which it was saved.

Note: The user of this parameter must have *ALLOBJ authority.

***NONE:** The differences described above are not allowed for the restore operation. For an ownership difference, the object is not restored. For an authorization list difference, the object is restored, but the object is not linked to the authorization list, and public authority is set to *EXCLUDE.

***ALL:** The differences described above are allowed for the restore operation. The object is restored.

Notes:

1. If the owners of the object do not match, the object is restored, but it retains the ownership and authorities it had before the restore operation.
2. The informational message causes a diagnostic message to be sent indicating that security or integrity changes occurred during the restore operation. This results in an escape message for the restore operation.
3. If the user is restoring objects to a system other than the one on which they were saved and the objects are secured by an authorization list, specifying *ALL automatically relinks the objects to the authorization list. If the authorization list does not exist on the new system, a message that includes the name of the missing list is issued.

OUTPUT

Specifies whether a listing that shows information about the status of the object is created and directed to an output file. The listing shows the restore information and shows all objects, restored, not restored, and excluded. Information about each object's security is listed for the restored objects. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***NONE:** No output is created.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: You must specify the database file name on the OUTFILE parameter when OUTPUT(*OUTFILE) is specified.

OUTFILE

Specifies the qualified name of the database file to which the information about the object is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QASRRSTO in QSYS with the format name QSRRST as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which the output of the command is directed.

OUTMBR

Specifies the name of the database file member to which the output is directed when OUTPUT(*OUTFILE) is specified.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that receives the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it. If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

Examples

Example 1: Restoring All Objects

```
RSTCFG  OBJ(*ALL)  DEV(TAP01)  OBJTYPE(*ALL)
```

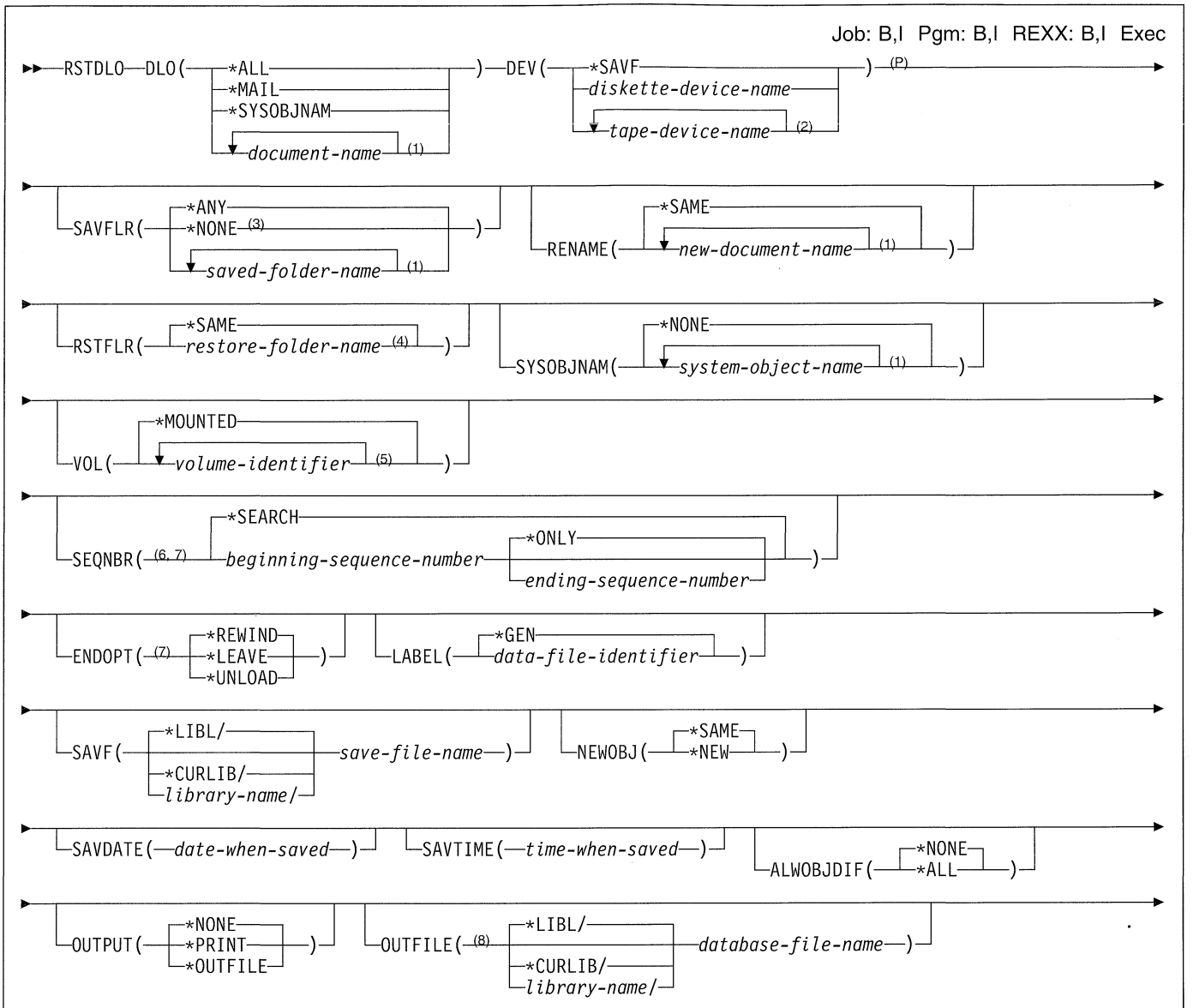
This command restores all of the device configuration and SRM objects from the tape on the TAP01 drive.

Example 2: Restoring a Device Description

```
RSTCFG  OBJ(PRT01)  DEV(TAP01)  OBJTYPE(*DEVD)
  VOL(ABCD)
```

The device description for PRT01 that was saved on tape volume ABCD is restored to the system. If device description PRT01 already exists on the system, it must be varied off before it can be restored.

RSTDLO (Restore Document Library Object) Command



Notes:

1 A maximum of 300 repetitions

2 A maximum of 4 repetitions

3 *NONE is valid only when DLO(*ALL) is specified.

4 This value is not valid when DLO(*SYSOBJNAM) is specified.

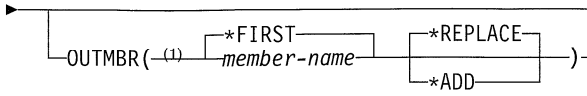
5 A maximum of 75 repetitions

6 For restore from diskette, or if SEQNBR(*SEARCH) is used to restore from tape, only the first file on diskette or tape that contains saved documents is restored.

7 Applies to tape devices only

8 This parameter is valid only when OUTPUT(*OUTFILE) is specified.

P All parameters preceding this point can be specified in positional form.



Note:
1 This parameter is valid only when OUTPUT(*OUTFILE) is specified.

Purpose

The Restore Document Library Object (RSTDLO) command restores documents, folders, and distribution objects (mail).

This command can be used to restore the documents and folders if the document was or was not freed by the Save Document Library Object (SAVDLO) command, or to restore documents and folders that were deleted by the Delete Document Library Object (DLTDLO) command.

Restoring a document either replaces the existing document content and control information if the document exists on the system, or it adds new document content and control information if the document does not exist.

For a filed document (electronic mail or a document stored in the document library in the SAA OfficeVision/400 program), the document and folder name of the document object on the media must be the same as the document name and folder name of the document on the system, unless the document is renamed and put in a different folder during the restore operation.

Note: Folder names must match exactly for restored folders. All objects that are not in use are restored from the folder on the media or in the save file to the existing folder. Restoring a folder creates a new folder object if the folder does not exist and adds to this new folder all objects saved with the folder on the media or in the save file. If the folder exists, any document or folder objects that do not exist within it are created. The existing documents are replaced with the version from the media.

For a filed document restored on the system whose owner is not known to the system or is not enrolled in the system distribution directory, the user profile of the default owner (QDFTOWN) becomes the owner of the document or folder.

The creation date of a document does not change if the document exists. If the document does not exist, the creation date is set to the date on which the document is created.

The security does not change if a document or folder exists on the system where it is to be restored. If the document or folder does not exist, public authority, authorization list, and personal status are restored; however, all other private document and folder authorities are not restored. These authorities must be established again by the owner.

If a document is restored that had a mail log entry when it was saved, the mail log entry is restored if the distribution tracking object exists on the system. If the distribution tracking object does not exist on the system, a message is sent saying that the document was restored without a mail log entry.

If this command ends abnormally, objects are left on the system in an unknown state and cannot be found in a library. This can happen if a power failure occurs when this command is run. The Reclaim Storage (RCLSTG) command can be used to clean up the auxiliary storage and delete most of those objects from the system; however, unknown mail objects are not cleaned up with the RCLSTG command.

When a set of documents and folders are restored, all documents and folders in the set must exist in the same diskette, tape, or save file.

If a document exists in more than one tape or diskette file, the user can control which document is restored by specifying the media file using the sequence number (tape) or label (tape or diskette) parameter. If more than one version of the document exists, the SAVDATE and SAVTIME parameters can also be used to select the correct document.

When text search services are on the system and the user restores a document library object, the text search index for the object is restored.

Restrictions:

1. To use this command, you must have *SAVSYS or *ALLOBJ special authority or be enrolled in the system distribution directory.
2. You cannot run another RSTDLO command, a SAVDLO command, or a Reclaim Document Library Object (RCLDLO) command specifying DLO(*ALL) while this command is running.

Required Parameters

DLO

Specifies the document library objects to be restored.

***ALL:** All documents, folders, and distribution objects (mail) that are saved on the media and meet the values specified on the SAVFLR parameter are restored.

***MAIL:** All distribution objects and documents that were referred to by a mail log are restored.

***SYSOBJNAM:** The documents with the system object names specified in the SYSOBJNAM parameter are restored.

document-name: Specify the user-assigned names of the documents to be restored. Up to 12 characters can be specified for the name; up to 300 document names can be specified. All documents named must be in the folder specified on the SAVFLR parameter.

DEV

Specifies the name of the device to be used to restore the document library objects. The device name must be known on the system by a device description.

***SAVF:** The document library objects are restored from a save file.

diskette-device-name: Specify the name of the diskette device to be used to restore the document library objects.

tape-device-name: Specify the names of one or more tape devices used for the restore operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. Using more than one tape device permits one tape volume to be rewound and unloaded while another tape device processes the next tape volume.

Optional Parameters

SAVFLR

Specifies the name of the folder on the media from which the documents and folders are restored.

***ANY:** All document library objects that meet the values specified on the DLO parameter are restored, regardless of the folders (if any) from which they were saved. This value is valid only if *ALL, *MAIL, or *SYSOBJNAM is specified on the DLO parameter.

***NONE:** The documents to be restored were saved as documents without folders.

Note: *NONE is valid only when DLO(*ALL) is specified.

saved-folder-name: Specify the name of the saved folder from which documents or folders are to be restored. Up to 63 characters can be specified for the folder name. When DLO(*ALL) is specified, up to 300 folder names can be specified. The name of a saved folder must be specified when DLO(*document-name*) is specified.

RENAME

Specifies the new user-assigned name for the restored document.

***SAME:** The documents being restored have the same user-assigned names that they have on the media.

new-document-name: Specify the new document names that the documents have after they are restored. Up to 300 user-assigned names can be specified for documents being restored.

RSTFLR

Specifies the name of the folder in which the folders and documents to be restored will be placed. The name can be any fully-qualified folder name up to 63 characters. The folder named is not required to exist if DLO(*ALL) is specified and the media contains the folder specified on the SAVFLR parameter.

***SAME:** The documents to be restored are placed into the same folder from which they were saved.

restore-folder-name: Specify the name of the folder in which the restored documents and folders are to be placed.

SYSOBJNAM

Specifies the system object name of the documents to be restored. This parameter is valid only when DLO(*SYSOBJNAM) is specified.

***NONE:** A system object name is not specified.

system-object-name: Specify the 10-character system object names of the documents to be restored. A maximum of 300 names can be specified.

VOL

Specifies the volume identifiers of the tape volumes or diskette volumes from which the objects are being restored. The volumes must be in the same order as they were when the objects were saved. The volume that contains the beginning of the file to be restored should be placed in the tape or diskette unit. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The objects are restored from the volumes placed in the device specified on the DEV parameter.

SEQNBR

Specifies the tape sequence numbers used for the restore operation. This parameter is valid for tape only.

***SEARCH:** The tape is searched for the first data file with an identifier matching the LABEL parameter value and with contents of a minimum of one of the specified document library objects. If the last operation on the device specified ENDOPT(*LEAVE) (that is, the tape is positioned at the location at which the last operation ended), the file search begins with the first data file beyond the current tape position. If ENDOPT(*LEAVE) was not specified on the last operation (or if the tape has been rewound since an ENDOPT(*LEAVE) operation), the search begins with the first data file on the volume.

Note: If a folder is contained in more than one file or on more than one tape, a message is sent indicating that the folder may not have been fully restored. To restore the rest of the folder, run the RSTDLO command again specifying the next sequence number.

Element 1: Beginning Sequence Number

Specifies the sequence number of the first file to be used for the restore operation.

Element 2: Ending Sequence Number

***ONLY:** The ending sequence number is the same as the beginning sequence number.

ending-sequence-number: Specify the sequence number of the last file used for the restore operation.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

LABEL

Specifies the file label that is used to find the file that was written to the media during the save operation.

***GEN:** The system generates the default name of the file label for which to search.

data-file-identifier: Specify the media data file identifier, which includes up to 17 alphanumeric characters, of the file that contains the documents and folders to be restored.

SAVF

Specifies the qualified name of the save file that contains the saved document library objects to be restored.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the save file from which to restore DLOs.

NEWOBJ

Specifies whether a new library-assigned name is created for the folder or document being restored.

***SAME:** The library-assigned name is restored to the name used when the document library objects were saved.

***NEW:** A new library-assigned name is created for each document or folder being restored. If *NEW is specified on this parameter, then ALWOBJDIF(*ALL) cannot be specified.

SAVDATE

Specifies the date on which the document library objects were saved. If more than one version of the document library objects exist on the media, use this parameter to identify which version of the document library objects to restore. The date must be expressed in the job date format; if separators are used, the value must be enclosed in apostrophes. If the SAVDATE parameter is not specified, the version of the documents and folders to be restored will be the first version found on the volume or the version found with the specified file label.

SAVTIME

Specifies the time when the document library objects were saved. If more than one version of the document library objects exist on the media with the same value for the date saved, use this parameter to identify which version of the document library objects to restore. Express the time as a 6-digit value, in the format hours, minutes, seconds (hhmmss). If separators are used, the value must be enclosed in apostrophes ('hh:mm:ss'). If a volume identifier is specified, but SAVTIME is not specified, the version of the document library objects to be restored will be the first version found on the volume or the version found with the specified file label.

This parameter is valid only if SAVDATE is also specified.

ALWOBJDIF

Specifies whether the following differences encountered during the RSTDLO operation are allowed:

- The owner of the object on the system is different than the owner of the object from the save.
- The system object name on the system does not match the system object name on the tape or diskette being restored.
- The object is secured by an authorization list and is being restored to a system other than the one on which it was saved.

The ALWOBJDIF parameter can be used to allow an object to be restored whose owner or object name on the system is different than on the media used for the restore operation. By specifying the *ALL special value, an object with a different name is restored to the name on the media, while an object with a different owner keeps the owner name from the system instead of the media.

Note: In order to use this parameter, *ALLOBJ authority is required.

***NONE:** The differences described above are not allowed on the restore operation. For authorization list cases, the object is restored, but the object is not linked to the authorization list, and public authority is set to

*EXCLUDE. For other cases, a diagnostic message is sent for the object, and the object is not restored.

***ALL:** All the differences described above are allowed for the restore operation. The object is restored.

Notes:

- If the owners of the object do not match, the object is restored, but it keeps the ownership and authorities of the object on the system before the restore operation.
- The informational message flags a diagnostic message to be sent indicating that security or integrity changes occurred during the restore operation. This results in an escape for the restore function.
- If *ALL is specified on this parameter, then NEWOBJ(*NEW) cannot be specified.
- If the user is restoring objects to a system different from the one on which they were saved and the objects are secured by an authorization list, specifying *ALL automatically links the objects to the authorization list. If the authorization list does not exist on the new system, a message that includes the name of the missing list is issued.

OUTPUT

Specifies whether a listing that shows information about the status of the object is created and directed to an output file. The listing shows the restore information and shows all objects, restored, not restored, and excluded. Information about each object's security is listed for the restored objects. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***NONE:** No output is created.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: You must specify the database file name on the OUTFILE parameter when OUTPUT(*OUTFILE) is specified.

OUTFILE

Specifies the qualified name of the database file to which the information about the object is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QAOJRSTO in QSYS with the format name QOJRST as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which the output of the command is directed.

OUTMBR

Specifies the name of the database file member to which the output is directed. If a member already exists, the system uses the second element of this parameter to determine whether the member is cleared before the new records are added. If the member does not exist and a member name is not specified, the system creates a member with the name of the output file specified on the OUTFILE parameter. If an output file member name is specified, but the member does not exist, the system creates it.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that receives the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it. If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

Examples

Example 1: Restoring Documents with System Object Names

```
RSTDLO  DLO(*SYSOBJNAM)  DEV(TAP01)
  SYSOBJNAM(HZ83B55219)
```

This command restores the document named HZ83B55219 from the tape unit TAP01.

Example 2: Restoring Documents from a Save Folder

```
RSTDLO  DLO(A)  DEV(diskette-device-name)  SAVFLR(X)
```

This command restores the document named A from folder X.

Example 3: Restoring All Documents

```
RSTDLO  DLO(*ALL)  DEV(TAP01)
```

This command restores all documents and folders that are on tape unit TAP01. If an object is on TAP01 more than once, each version of it is restored. The last version restored is the final version.

Example 4: Creating New Library-Assigned Name

```
| RSTDLO  DLO(*SYSOBJNAM)  DEV(TAP01)
    SYSOBJNAM(HZ83B55219)  NEWOBJ (*NEW)
```

| This command restores document HZ83B55219 from tape
| unit TAP01 and gives it a new library-assigned name and a new system object name.

Example 5: Renaming Documents

```
| RSTDLO  DLO(A B)  DEV(TAP01) SAVFLR(C)  RENAME(Y  Z)
    RSTFLR(X)
```

This command restores documents A and B from within folder C. Document A is renamed to Y and document B is renamed to Z. The command then puts them in folder X.

Example 6: Specifying Sequence Numbers

```
RSTDLO  DLO(*ALL)  DEV(tape-device-name) SAVFLR(A)
  SEQNBR(1 3)  LABEL(*GEN)
```

| This command restores all of folder A from tape files with the
| sequence numbers 1, 2, and 3, and the label QDOC.

Example 7: Specifying Allowed Differences

```
| RSTDLO  DLO(A)  DEV(TAP01)  SAVFLR(X)
    ALWOBJDIF(*ALL)
```

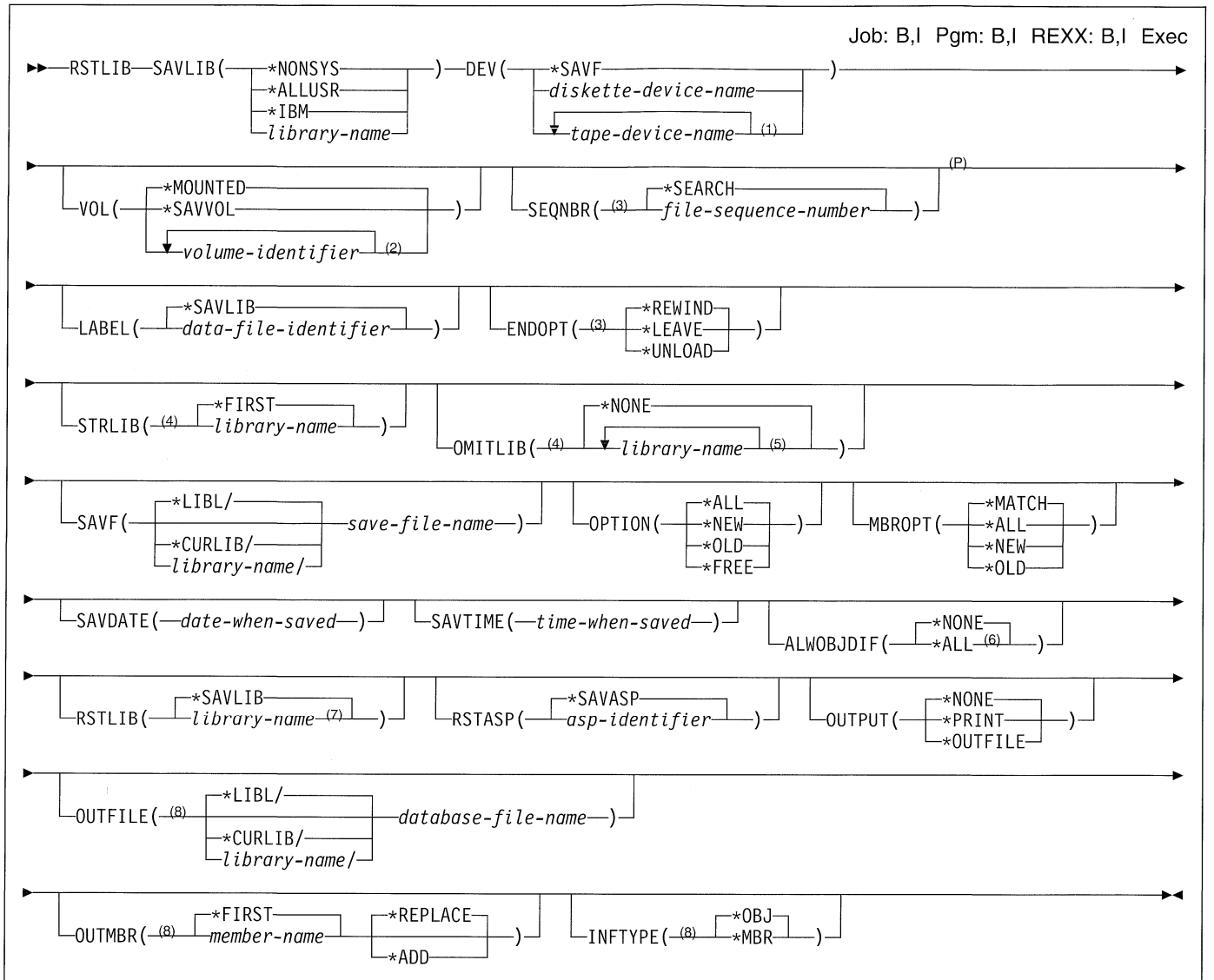
This command restores document A from folder X. If docu-
| ment A in folder X exists on the system and the owner of the document on the system does not match the owner of the document being restored, the document is restored and the owner of the document on the system remains unchanged.

| Example 8: Reporting Information about Objects | Restored and Not Restored

```
| RSTDLO  DLO(*ALL) DEV(TAP01) OUTPUT(*OUTFILE)
|   OUTFILE(INFO92) OUTMBR(FOURQT *ADD)
```

| This command restores all documents and folders from the
| tape device TAP01. A list reporting information about objects
| restored and objects not restored is directed to the output file
| INFO92. The output is received in the member FOURQT as
| an addition to existing information in the member.

RSTLIB (Restore Library) Command



Notes:

1 A maximum of 4 repetitions

2 A maximum of 75 repetitions

3 Applies to tape devices only

4 Valid only if SAVLIB(*NONSYS), SAVLIB(*ALLUSR), or SAVLIB(*IBM) is specified.

5 A maximum of 300 repetitions

6 When using ALWOBJDIF(*ALL), the MBROPT(*MATCH) parameter cannot be specified.

7 If an SQL database is restored to a library other than the one in which it was saved, the journals are not restored.

8 This parameter is valid only if OUTPUT(*OUTFILE) is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Restore Library (RSTLIB) command restores to the system one or a group of libraries saved on diskette or tape or it restores a user library saved in a save file. Any library that was saved by the Save Library (SAVLIB) command can be restored by this command. The RSTLIB command restores the whole library, which includes the library description, object descriptions, and contents of the objects in the library.

For job queues, message queues, output queues, data queues, and logical files, only the object descriptions are restored, because only the definitions are saved. Also, logical file access paths may be restored if they were saved. More information on restoring access paths is in the *Database Guide*.

This command can be used to restore libraries whose objects had their storage freed by the corresponding SAVLIB command of the restore operation, or libraries deleted by the Delete Library (DLTLIB) command. If the data portions of the objects in the saved libraries were not freed, each library is copied into the same area of storage that it previously occupied. If the storage was freed, the system finds the needed storage to store the library contents (the object description and data portion of every file, module, program, service program, Structured Query Language (SQL) package, and journal receiver in the library). If the library does not exist on the system because it has been deleted or is being restored on a different system, the system must find the storage for everything that is in the library, including the library description.

When the owner profile does not exist on the system, the user profile of the system default owner (QDFTOWN) becomes the default owner of any object being restored in the system.

If an object already exists in the library in which it is being restored, the public and private authorities of the existing object are retained. If the object does not exist in the library, all public authorities are restored, but private authorities must be granted again. For an existing output queue object that is actively spooling during the restore operation, or a data queue that already exists in the library, the object is not restored, and a diagnostic message is sent.

If an object is being restored over an existing object on the system, the object auditing value of the existing object is kept. If the object is being restored as new to the system, the object auditing value is restored from the media. Additionally, if the object is a library, the default auditing value for each object created in the library is restored if the library is being restored as new; otherwise, the default auditing value is restored from the media.

Notes:

1. To restore a save file to a library where it does not already exist, a user must have authority to the Create Save File (CRTSAVF) command.
2. The RSTLIB command ignores all file overrides currently in effect for the job, except the overrides for the restore output file.

Restrictions:

1. The user of this command must have *SAVSYS authority specified in the user profile by the SPCAUT parameter, or must have all of the following:
   a. Read and add authority for, or be the owner of, each library specified.
   b. Object existence authority for, or be the owner of, each object in the library if the object already exists in the library on the system. Object existence and use authority are required for message queue objects. If the object does not exist, the user must have add authority for the user profiles that own each object being restored. If the user does not have the correct authority for all the libraries and objects specified, only those for which the authority has been given are restored.
   c. If VOL(*SAVVOL) is specified, use authority to the saved-from library.
   d. Use authority for the save file is required when restoring libraries from a save file. Use authority for the device description and the device file are required when restoring libraries from tape or diskette.
2. The current versions of programs on the system should not be run while the library that contains those programs is being restored. If any program is running while it is being restored, the program will not be restored.

Required Parameters

SAVLIB

Specifies the name of the library, or the *NONSYS, *ALLUSR, or *IBM group of libraries, to be restored on the system. You cannot specify *NONSYS, *ALLUSR, or *IBM if RSTLIB(*SAVLIB) is not specified.

***NONSYS:** Libraries saved by the Save Library (SAVLIB) command with LIB(*NONSYS) specified are restored. All other operations on the system must be ended before this option is specified. This requires ending all subsystems through the End Subsystem (ENDSBS(*ALL)) command or End System (ENDSYS) command.

You can do a RSTLIB SAVLIB(*IBM) and a RSTLIB SAVLIB(*ALLUSR) from a SAVLIB LIB(*NONSYS).

***ALLUSR:** All user libraries are restored. All libraries with names that do not begin with the letter Q are restored except for the following:

```
#CGULIB     #DSULIB     #SEULIB
#COBLIB     #RPGLIB
#DFULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered "user libraries", and are also restored:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

***IBM:** Restores all system (IBM) libraries except for the following:

```
QDOC        QPFRDATA    QSPL        QTEMP
QDSNX       QRCL        QSRV        QUSER38
QGPL        QRECOVERY   QSYS        QUSRSYS
QGPL38      QRPLOBJ     QS36F       QUSRVxRxMx
```

Note: A different library name, in the format QUSRVxRxMx, may be created by the user for each previous release supported by IBM to contain any user commands to be compiled in a CL program for the previous release. For the QUSRVxRxMx user library, VxRxMx is the version, release, and modification level of a previous release that IBM continues to support.

The following libraries with names that do not begin with the letter Q are also restored:

```
#CGULIB     #DSULIB     #SEULIB
#COBLIB     #RPGLIB
#DFULIB     #SDALIB
```

library-name: Specify the name of the library to restore. The library name must be the same name that was used when the library was saved. The names QSYS, QSRV, QTEMP, QSPL, QDOC, QRPLOBJ, and QRECOVERY cannot be specified.

DEV

Specifies the name of the device used for the restore operation. The device name must already be known on the system by a device description.

***SAVF:** The restore operation is done by using the save file specified on the SAVF parameter.

diskette-device-name: Specify the name of the diskette device that is used for the restore operation.

tape-device-name: Specify the names of one or more tape devices used for the restore operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. Using more than one tape device permits one tape volume to be rewound and unloaded while another tape device processes the next tape volume.

Optional Parameters

VOL

Specifies the volume identifiers of the tape volumes or diskette volumes from which the objects are being restored. The volumes must be in the same order as they were when the library was saved. If the library was known to the system before the restore operation, the system can verify whether the volumes put on tape contain the current version of the library. The volume that contains the beginning of the file to be restored should be placed in the tape or diskette unit. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The library is restored from the volumes that are currently placed in the device specified on the DEV parameter. The first version of the saved library that is found on the media is restored, unless a specific version is identified by the SAVDATE parameter and SAVTIME parameter, or for tape, the SEQNBR parameter.

***SAVVOL:** The system, by using the save or restore history information, determines which tape or diskette volumes contain the most recently saved version of the library. If the device type specified in the DEV parameter does not match the device type of the most recently saved version of the library, an error message is sent to the user, and the function is ended. If *SAVVOL is specified, the parameters SAVDATE and SAVTIME cannot be specified. IF *SAVVOL is specified, *SEARCH must be specified on the SEQNBR parameter.

volume-identifier: Specify the identifiers of one or more volumes in the order they are placed in the device and used to restore the libraries.

SEQNBR

Specifies, only when tape is used, which sequence number is used for the restore operation.

***SEARCH:** The tape is searched for a data file with an identifier that matches the LABEL parameter value; when a match is found, the data file is restored. If the last operation on the device specified ENDOPT(*LEAVE), the tape is positioned at the location where the last operation ended, and the file search starts with the first data file beyond the current tape position. If ENDOPT(*LEAVE) was not used for the last operation, or if the tape was manually rewound after an ENDOPT(*LEAVE) operation, the search starts with the first data file on the volume.

file-sequence-number: Specify the sequence number of the file that is used.

If *NONSYS, *ALLUSR, or *IBM is specified on the SAVLIB parameter, the sequence number specifies the location of the file QFILE. The QFILE file is at the beginning of the *NONSYS, *ALLUSR, or *IBM save operation. The QFILE file contains the list of libraries saved.

LABEL

Specifies the name that identifies the data file on the tape or diskette used for the restore operation. This must be the label specified on the save command.

***SAVLIB:** The file label is the name of the library specified on the SAVLIB parameter.

data-file-identifier: Specify the data file identifier (up to 17 characters) of the data file used for the restore operation. This option is valid only for single library restore operations.

ENDOPT

Specifies, when tape is used, what positioning operation is automatically done on the tape volume after the restore operation ends. If more than one volume is included, this parameter applies only to the last volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

STRLIB

Specifies the name of the starting library for a *NONSYS, *IBM, or *ALLUSR restore operation.

If an irrecoverable media error occurs during the restore operation, this parameter can be used to restart the operation.

This parameter is valid only when *NONSYS, *IBM, or *ALLUSR is specified for the restore operation.

Note: In the recovery steps that follow, *NONSYS is specified on the SAVLIB parameter. If you are restoring IBM-supplied libraries or all user-created libraries and IBM-supplied libraries, specify *IBM or *ALLUSR instead.

The basic recovery steps for a restore operation are:

1. Look at the job log to determine the library where the previous restore library (RSTLIB SAVLIB(*NONSYS)) command failed. Find the last library restored, which is indicated by a successful restore completion message.
2. Load the first tape of the SAVLIB LIB(*NONSYS) media.
3. Issue the following command:

   ```
   RSTLIB   SAVLIB(*NONSYS)  DEV(tape-name)
     ENDOPT(*LEAVE)  STRLIB(library-name)
     OMITLIB(library-name)
   ```

   where the *library-name* for the STRLIB and OMITLIB parameters is the library where the RSTLIB failed. This starts the restore operation on the library after the library where the restore operation failed.
4. When you are prompted, load the volume containing the starting library.
5. After the restore operation is complete, restore the library where the restore operation failed using the media from a previous save operation.

   Note: Consider removing the tape with the media error from the next save rotation cycle to avoid another tape error.

***FIRST:** The restore operation begins with the first library saved.

library-name: Specify the name of the library where the restore operation begins.

OMITLIB

Specifies a list of libraries to be excluded from the restore operation.

Note: This parameter is valid only when *NONSYS, *IBM, or *ALLUSR is specified on the SAVLIB parameter.

***NONE:** No libraries are excluded from the restore operation.

library-name: Specify the names of the libraries to be excluded from the restore operation.

SAVF

Specifies the qualified name of the save file used to restore the save data.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the file from which to perform the restore operation.

OPTION

Specifies how to handle restoring each object.

***ALL:** All the objects in the saved library are restored to the library. Old objects on diskette, tape, or in a save file replace the current versions in the library on the system, and the objects not having a current version are added to the library on the system. Objects presently in the library but not on the media, remain in the library.

***NEW:** Only the objects in the saved library that do not exist in the current version of the library on the system are added to the library. Only objects not known to the library on the system are restored; known objects are not restored. This option restores objects that were deleted after they were saved or that are new to this library. If any saved objects have a version already in the library on the system, they are not restored, an informational message is sent for each object, and the restore operation continues.

***OLD:** Only the objects in the library having a saved version are restored; that is, the version of each object currently in the library is replaced by the saved version. Only objects known to the library are restored. If any saved objects are no longer part of the online version of the library, they are not added to the library and an informational message is sent for each object, but the restore operation continues.

***FREE:** The saved objects are restored only if they exist in the library on the system with their space freed. The saved version of each object is restored on the system in its previously freed space. This option restores objects that had their space freed when they were saved. If any saved objects are no longer part of the current version of the library, or if the space is not free for any object, the object is not restored and an informational message is sent for each object. The restore operation continues, and all of the freed objects are correctly restored.

MBROPT

Specifies, for database files currently on the system, which file members are restored.

Note: If MBROPT(*MATCH) is used, the member list in the saved file must match, member for member, the current version in the system. All members are restored for files that do not exist. Before restoring a file, the system verifies whether the file and member creation dates of the known system objects match the creation dates of the objects being restored. If verification fails, the file is not restored.

***MATCH:** The saved members are restored if the lists of the members where they exist match, member for member, the lists of the current system version. *MATCH cannot be specified if *ALL is specified on the ALWOBJDIF parameter.

***ALL:** All members in the saved file are restored.

***NEW:** Only new members (members not known to the system) are restored.

***OLD:** Only members already known to the system are restored.

SAVDATE

Specifies the date the library was saved. If the most recently saved version is not restored, or if more than one saved version is on the volumes, enter the date that specifies which version of the library is restored. The date must be entered in the job date format. If separators (specified by the system value QDATSEP) are used, the value must be enclosed in apostrophes. If a volume identifier or VOL(*MOUNTED) is specified, but SAVDATE is not, the first version of the library found is restored. This parameter is valid only if a volume identifier is specified, VOL(*MOUNTED) is specified, or if a save file (SAVF parameter) is specified. Specify the date when the file was saved.

SAVTIME

Specifies the time when the library was saved, if the current version is not restored. The time can be specified with or without a time separator:

- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds.
- With a time separator, specify a string of 5 or 8 digits where the time separator specified for your job is used to separate the hours, minutes, and seconds. If you enter this command from the command line, the string must be enclosed in apostrophes. If a time separator other than the separator specified for your job is used, this command will fail.

If a volume identifier or *MOUNTED is specified on the VOL parameter, but this parameter is not, the first version of the library found on the volume is restored. This parameter is valid only if the SAVDATE parameter is also specified.

This parameter is valid only if SAVDATE is also specified.

ALWOBJDIF

Specifies whether certain differences encountered during a restore operation are allowed. The differences include:

- Ownership: The owner of an object on the system is different than the owner of an object from the save operation.
- File creation date: The creation date of the database file on the system does not match the creation date of the file that was saved.
- Member creation date: The creation date of the database file on the system does not match the creation date of the member that was saved.
- Validation value verification: The validation value created at the time an object was created does not match the validation value created during the restore operation of an object on a system with a QSECURITY level of 40 or higher.
- Authorization list linking: The object is being restored to a system different from the one on which it was saved.

Note: The user of this parameter must have *ALLOBJ authority. If *ALL is specified, *MATCH cannot be specified on the MBROPT parameter.

***NONE:** None of the differences described above are allowed on the restore operation. For validation value verification failure cases, the object is restored but ownership is transferred to QDFTOWN and all authorities are revoked. For authorization list cases, the object is restored, but the object is not linked to the authorization list, and public authority is set to *EXCLUDE. For all

other cases, a diagnostic message is sent for the object, and the object is not restored.

***ALL:** All of the differences listed above are allowed for the restore operation. An informational message is sent, except for validation value verification and authorization list cases, and then the object is restored. The following facts should be noted:

- If the owners of the object do not match, the object is restored, but the ownership and authorities it had before the restore operation are retained.
- Either the file itself or members of the file are restored based on the following facts:
  1. If there is a file-level mismatch and ALWOBJDIF(*ALL) has been specified, the existing version of the file is renamed and the saved version of the file is restored.
  2. If there is a member-level mismatch, the existing version of the member is renamed and the saved version of the member is restored.
- Validation value verification:
  1. For programs with no validation value stored on the media (in reference to program data saved on an OS/400 system prior to version 1 release 3.0), the program is restored and the system does not attempt to recreate it. Nothing is logged in the security journal and no informational messages are sent.
  2. For programs that do have a validation value stored on the media and the system fails to recreate the program, the original version of the program is restored and an entry is logged in the security journal and job log.
- The informational message causes a diagnostic message to be sent indicating that security or integrity changes occurred during the restore operation. This results in sending an escape message for the restore operation.
- If the user is restoring objects to a system different from the one on which they were saved and the objects are secured by an authorization list, specifying *ALL automatically links the objects to the authorization list again. If the authorization list does not exist on the new system, a message that includes the name of the missing list is issued and the public authority is set to *EXCLUDE.

RSTLIB

Specifies whether the library contents are restored to the same library in which they were saved, or to a different library. If a different library is specified, the user cannot specify *NONSYS, *ALLUSR, or *IBM on the SAVLIB parameter.

***SAVLIB:** The library contents are restored to the same library or libraries in which they were saved.

library-name: Specify the name of the library where the saved library contents are being restored. If *NONSYS, *ALLUSR, or *IBM is specified on the SAVLIB parameter, a library name cannot be specified on this parameter.

Note: If an SQL database is restored to a library other than the one in which it was saved, the journals are not restored.

RSTASP

Specifies whether the libraries and objects are restored to the same auxiliary storage pool (ASP) from which they were saved or to another specific ASP. ASP 1 is the system ASP. User ASPs can be configured as ASPs 2 through 16.

A user ASP can contain either:

- Libraries and all objects in the libraries.
- Journals, journal receivers, and save files for libraries in the system ASP.

This parameter applies to all libraries that are specified. More information about user ASPs is in the *Advanced Backup and Recovery Guide*.

Warning!

System or product libraries (libraries that begin with a Q or #) must not be created in or restored to a user ASP. Doing so can cause unpredictable results.

***SAVASP:** The objects are restored to the same auxiliary storage pools from which they were saved.

asp-identifier: Specify the ASP identifier. When the specified ASP is 1, the objects are restored to the system ASP, and when the specified ASP is 2 through 16, the objects are restored to the user-defined ASP specified by the identifying number.

OUTPUT

Specifies whether a listing that shows information about the status of the object is created and directed to an output file. The listing shows the restore information and shows all objects, restored, not restored, and excluded. Information about each object's security is listed for the restored objects. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***NONE:** No output listing is created.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: You must specify a database file name on the OUTFILE parameter when OUTPUT(*OUTFILE) is specified.

OUTFILE

Specifies the qualified name of the database file to which the information is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QASRRSTO in QSYS with the format name QSRRST as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which output from the command is directed. If this file does not exist, it is created in the specified library. More information on defining the format of database files as output files is in the *Database Guide*.

OUTMBR

Specifies the name of the database file member to which the output is directed. This parameter is valid only if OUTPUT(*OUTFILE) is specified.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

INFTYPE

Specifies the type of information directed to the database file. This parameter is valid only if OUTPUT(*OUTFILE) is specified.

***OBJ:** The list contains an entry for each object requested to be restored.

***MBR:** The list contains an entry for each object or, for database files, each member requested to be restored.

Examples

Example 1: Restoring New Objects

```
RSTLIB  SAVLIB(JOE)  DEV(TAP01)  OPTION(*NEW)
```

This command restores the saved version of library JOE from tape device TAP01. The only objects that are restored in the library are new objects (ones that were in the library when they were saved and later deleted).

Example 2: Printing Output

```
RSTLIB  SAVLIB(*NONSYS)  DEV(TAP01)  OUTPUT(*PRINT)
```

This command restores all the saved user-created libraries, the QGPL library, and the licensed program libraries (such as QRPG and QIDU) to the system from tape. The contents of the libraries are restored exactly as they were saved. New objects (on tape) are added to the system; old objects in the system are overlaid by the version of the old objects on tape. Because OUTPUT(*PRINT) is specified, a printout of all objects (restored and not restored) for each library, is sent to the printer with the job's spooled output. Each library after the first library starts on a new page. After each library, a completion message states how many objects were restored and how many were not restored. At the end of a list, a final completion message states how many libraries were restored and how many were not restored.

Example 3: Specifying Where the Restore Operation Begins

```
RSTLIB  SAVLIB(*NONSYS)  DEV(TAP01)  STRLIB(MIKESLIB)
```

This command restores the saved nonsystem libraries beginning with library MIKESLIB from the tape device, TAP01. The nonsystem libraries are saved in alphabetic order. Therefore, all libraries beginning with and following the name MIKESLIB are restored. The *first* tape of the nonsystem save must be loaded. An inquiry message instructs the user to load the tape containing MIKESLIB. If necessary, the same message is sent until the tape containing MIKESLIB is found.

Example 4: Restoring a Version From a Specific Date and Time

```
RSTLIB  SAVLIB(PAYROLL)  DEV(TAP01)  SAVDATE(060193)
  SAVTIME(103214)  RSTLIB(OLDPAY)  VOL(PAY)
```

This command restores the version of the PAYROLL library from the device TAP01, whose volume identifier is PAY. The version to be restored was saved at 10:32:14 on the date 06/01/93. All of the objects in the saved PAYROLL library are restored to the library OLDPAY. All new files are restored. Old files are restored only if the member lists of the files on the tape match the member lists of the files on the system.

Example 5: Restoring From Multiple Tape Volumes

```
RSTLIB  SAVLIB(QGPL)  DEV(TAP01) VOL(QGPL QGPL)
```

This command restores the QGPL library from two tape volumes both named QGPL. Even though the volume identifiers are the same, they must both be specified.

Example 6: Restoring From Multiple Tape Devices

```
RSTLIB  SAVLIB(USRLIB)  DEV(TAP01 TAP02 TAP03)
  VOL(USRA USRB USRC USRD)  ENDOPT(*UNLOAD)
```

This command restores library USRLIB from four volumes on three tape devices. Volume USRA is put on tape device TAP01, volume USRB on TAP02, volume USRC on TAP03, and volume USRD on TAP01. The operator removes volume USRA from TAP01, so that TAP01 can be used by volume USRD. If the tape volumes are put in the wrong order, an error message is sent to the system operator message queue.

Example 7: Restoring a Specific Version

```
RSTLIB  SAVLIB(LIB1)  DEV(TAP01)  MBROPT(*ALL)
  SAVDATE(082392)  SAVTIME(123251)
  RSTLIB(LIB2)  OUTPUT(*PRINT)
```

This command restores the version of library LIB1 from the device TAP01. The version to be restored was saved at 12:32:51 on the date 08/23/92. All of the objects in the saved library LIB1 are restored to library LIB2. A list of restored objects and those not restored is given. All files and file members are restored.

Example 8: Restoring a Library From a Save File

```
RSTLIB  SAVLIB(LIB1)  DEV(*SAVF)  SAVF(SAVF1)
```

This command restores library LIB1 from the save file SAVF1.

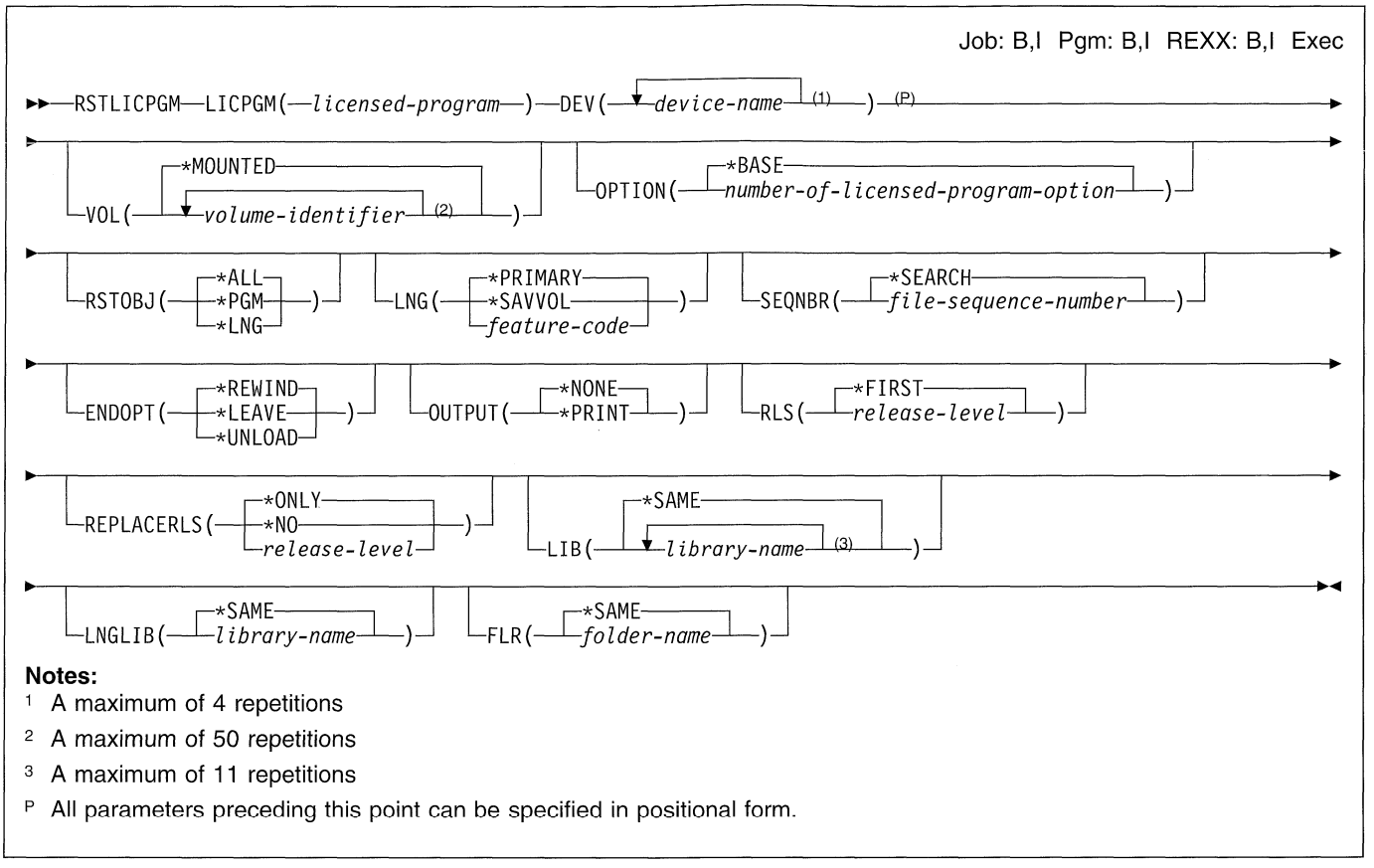
Example 9: Restoring to a User-Defined ASP

```
RSTLIB  SAVLIB(LIB1)  DEV(*SAVF)  SAVF(SAVF1)
  RSTASP(2)
```

This command restores the library named LIB1 from the save file named SAVF1. The library and all objects in the saved version of LIB1 are restored to ASP 2 unless:

- The library already exists in a different ASP.
- ASP 2 contains a journal, journal receiver, or SAVF which is part of a library in the system ASP.
- ASP 2 does not exist on the system.
- There are object types in the library which cannot be restored to user ASPs. These objects will not be restored.

RSTLICPGM (Restore Licensed Program) Command



Notes:

1 A maximum of 4 repetitions

2 A maximum of 50 repetitions

3 A maximum of 11 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Restore Licensed Program (RSTLICPGM) command loads or restores a licensed program, either for initial installation or new-release installation.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. To use this command, the user must have *SECADM and *ALLOBJ authority.
3. If this command is used to restore a program in the licensed program, the copy of the program currently in the system should not be running while the program is being restored. If this occurs, the processing program is ended abnormally.
4. If other objects of the licensed program are in use, they are not restored.
5. With the exception of overrides for the restore operation printing OUTPUT(*PRINT), this command ignores all file overrides currently in effect for the job.
6. Some licensed programs are restored only if the user is enrolled in the system distribution directory. See the publications for each licensed program for a description of this restriction.
7. This command does not restore code and language objects for the base OS/400 system.
8. This command does not support the use of user ASPs (auxiliary storage pools). All objects supplied by a licensed program must remain in the system ASP.

Required Parameters

LICPGM

Specifies the 7-character identifier of the licensed program being restored.

A list of IBM-supplied licensed programs is in the *Licensed Programs and New Release Installation Guide*.

DEV

Specifies the name of the tape device used for restoring the licensed program. The device name must be known by a device description on the system. If multiple devices are specified, they must have compatible media formats. Up to four device names can be specified.

Use the Work with Device Descriptions (WRKDEVD) command to display the names of the tape devices available on this system.

Optional Parameters

VOL

Specifies the volume identifiers of the tape volumes or diskette volumes from which the objects are being

restored. The volumes must be in the same order as they were when the library was saved. If the library was known to the system before the restore operation, the system can verify whether the volumes put on tape contain the current version of the library. The volume that contains the beginning of the file to be restored should be placed in the tape or diskette unit. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The licensed program is restored from the volumes that are mounted on the device specified on the DEV parameter.

volume-identifier: Specify the identifiers of one or more volumes in the order they are mounted on the device and used to restore the licensed program.

OPTION

Specifies which licensed program option to restore.

***BASE:** Only the base part of the licensed program is restored.

number-of-licensed-program-option: Specify the number associated with the optional part of the listed licensed program being restored.

RSTOBJ

Specifies the type of licensed program objects being restored.

***ALL:** All objects of the licensed program are restored. This includes both the program objects and the language objects.

The RSTOBJ(*ALL) value is used when the saving of the licensed program has been done with the SAVLICPGM command such that the language objects follow the program objects on the media. If the language objects (*LNG) and programming objects (*PGM) are not in consecutive order on the distribution tape, *ALL cannot be used in most cases. Instead, the program and language objects must be restored separately. The DSPTAP command can be used to determine the order of the objects on the tape.

***PGM:** Only the program objects for the licensed program are restored. This value is used when restoring the program objects from the distribution media where the program objects and the selected language objects are not on the same tape or are not in consecutive order.

***LNG:** The objects associated with the language identified by the language feature indicated on the LNG parameter are restored.

LNG

Specifies the national language version (NLV) to be used for restoring the licensed program. If the language feature of the licensed program on the save media matches the system language feature, the language objects are restored to the licensed program's libraries. If the language features do not match, the language objects are restored into the multilingual library for that language feature.

***PRIMARY:** The language feature of the operating system is restored for the specified product.

Note: Use the GO LICPGM function with option 20 to display the primary language of the operating system.

***SAVVOL:** The language file on the mounted volume is to be restored for the licensed program specified by the user.

feature-code: Specify the NLV identifier for the language file that is to be restored for the licensed program specified by the user. More information on feature identifications and a list of the IBM-supplied feature codes is in the *Licensed Programs and New Release Installation Guide*.

SEQNBR

Specifies which sequence number to use for the restore operation.

***SEARCH:** The volume on the device is searched for a data file for the licensed program; when a match is found, the objects are restored. If the last operation on the device specifies ENDOPT(*LEAVE), indicating that the tape is positioned at the location where the last operation ended, the file search starts with the first data file beyond the current tape position. If ENDOPT(*LEAVE) was not used for the last operation, or if the tape was manually rewound since an ENDOPT(*LEAVE) operation, the search starts with the first data file on the volume.

file-sequence-number: Specify the sequence number of the file used for the restore operation.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is automatically rewound, but not unloaded, after the restore operation has ended.

***LEAVE:** The tape is not rewound. Another restore operation can start at the current position on the tape.

***UNLOAD:** The tape is automatically rewound and unloaded after the restore operation has ended.

OUTPUT

Specifies whether output showing information about the status of the licensed program is created. The output shows all restored and unrestored objects. Information relative to each object's security is listed for the restored objects.

***NONE:** No output is created.

***PRINT:** The output is printed with the job's spooled output.

RLS

Specifies the version, release, and modification level of the product being restored.

***FIRST:** The first version, release, and modification level found on the tape is restored.

release-level: Specify the release level in the format VxRxMy, where Vx is the version number, Rx is the release number, and My is the modification number. Valid values for *x* are the numbers 0 through 9. Valid values for *y* are the numbers 0 through 9 and the letters A through Z.

REPLACERLS

Specifies the version, release, and modification level of the product being replaced.

***ONLY:** Replace only the version, release, and modification level of the product currently installed.

***NO:** The product currently installed on the system is not replaced. The product being restored must be a different release than the product currently installed. If the product being restored exists in the same libraries as the installed product, then an override parameter must be specified indicating to which libraries the product is restored.

release-level: Specify the release level in the format VxRxMy, where Vx is the version number, Rx is the release number, and My is the modification number. Valid values for *x* are the numbers 0 through 9. Valid values for *y* are the numbers 0 through 9 and the letters A through Z.

LIB

Specifies the libraries into which the product is being restored. This function is not supported by all products.

***SAME:** The product is restored into the specified library.

library-name: Specify the name of the library into which the product is being restored.

LNGLIB

Specifies the secondary language library into which the product is being restored. This function is not supported by all products.

***SAME:** The product is restored into the specified secondary language library.

library-name: Specify the name of the secondary language library into which the product is restored.

FLR

Specifies the name of the root folder into which the product is being restored. This function is not supported by all products.

***SAME:** Use the specified root folder.

folder-name: Specify the name of the root folder. The root folder is the folder on the system containing all of the other folders.

Examples

Example 1: Restoring Program Using Defaults

```
RSTLICPGM  LICPGM(5738BA1)  DEV(TAP01)
```

This command restores the BASIC licensed program to the system. The tape containing the licensed program objects must be put on the TAP01 tape drive. Because no other parameters are specified, the defaults are used for the command.

Example 2: Restoring a Second VRM of a Product

```
RSTLICPGM  LICPGM(vendor1)  OPTION(*BASE)  DEV(TAP01)
  RLS(V2R2M0)  REPLACERLS(*NO)  LIB(A B C)
```

This command restores the *BASE option of the V2R2M0 vendor1 product to the system if the *BASE option of the V2R2M0 vendor1 product is not currently installed on the system.

If no version release modification level of the *BASE option of the vendor1 product is installed on the system, the REPLACERLS parameter is ignored and the vendor1 product is installed on the system into libraries A, B and C.

If a version release modification level of the *BASE option of the vendor1 product is installed on the system but the *BASE option of the vendor1 product does not reside in the A, B and C libraries the V2R2M0 vendor1 product is restored into those libraries.

If any version release modification level of the *BASE option of the vendor1 product resides in the A, B and C libraries the V2R2M0 vendor1 product is not restored since the REPLACERLS parameter is set to *NO.

Example 3: Restoring One VRM of a Product Over Another VRM

```
RSTLICPGM  LICPGM(vendor2)  OPTION(*BASE)  DEV(TAP01)
  RLS(*FIRST)  REPLACERLS(*ONLY)
```

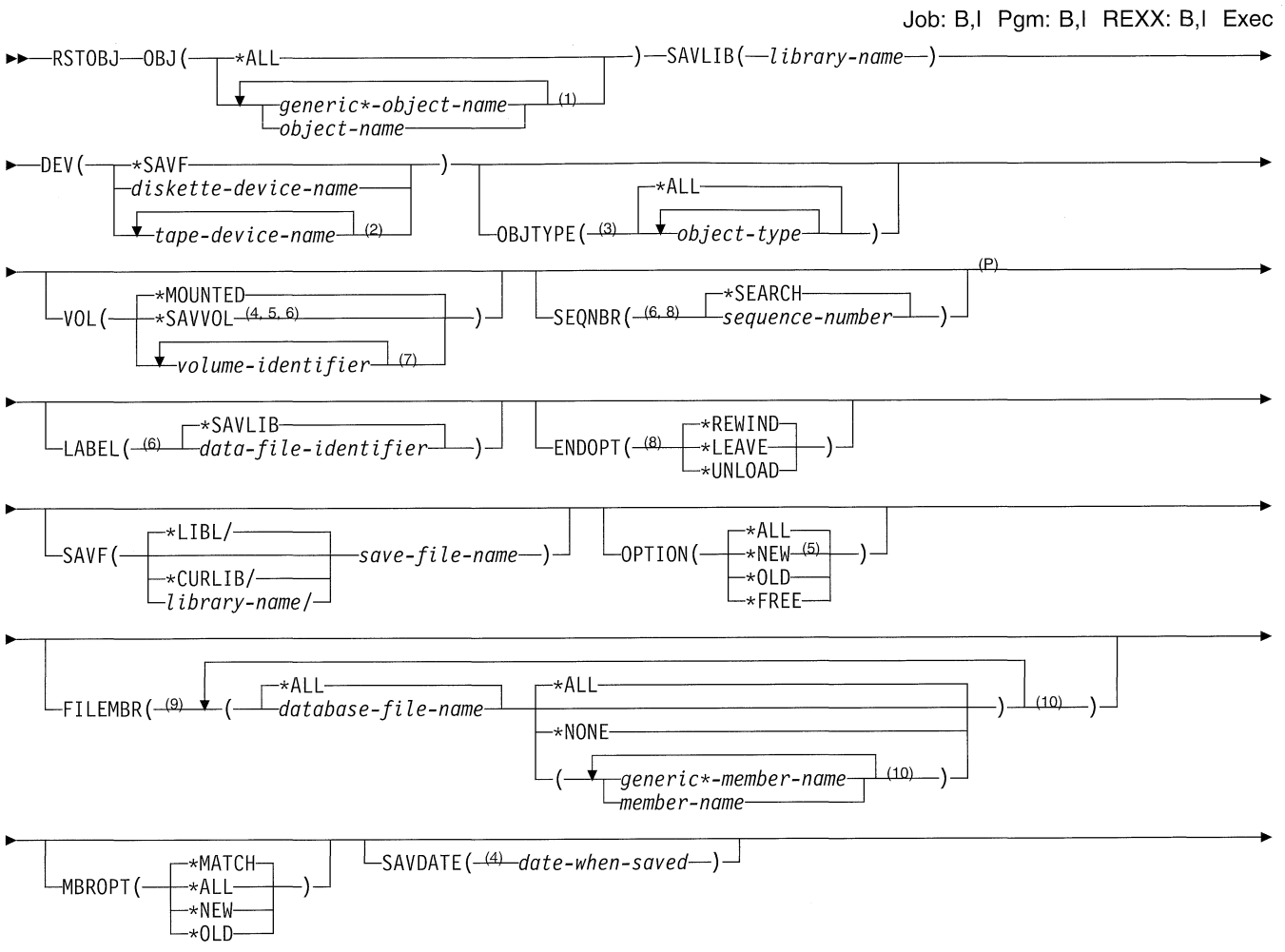
This command restores the first version release modification level of the *BASE option of the vendor2 product that is found on the tape in the TAP01 drive. It does not matter if the version release modification level of the *BASE option of the product on the system matches the version release modification level of the *BASE option of the product on the system.

If more than one version release modification level of the *BASE option of the vendor2 product currently exists on the system, the command is ended without restoring the specified product.

If only one version release modification level of the *BASE option of the vendor2 product currently exists on the system, the product from the tape is restored.

| The *BASE option of the product is restored into the library
| or libraries in which the *BASE option of the product on the
| system currently resides.

RSTOBJ (Restore Object) Command



Notes:

1 A maximum of 300 repetitions

2 A maximum of 4 repetitions

3 A list of the valid OS/400 object types for this command is in Appendix A.

4 If VOL(*SAVVOL) is specified, SAVDATE and SAVTIME cannot be specified.

5 If VOL(*SAVVOL) and RSTLIB(*SAVLIB) are specified, OPTION(*NEW) cannot be specified.

6 If VOL(*SAVVOL) is specified, only SEQNBR(*SEARCH) and LABEL(*SAVLIB) can be specified for these values.

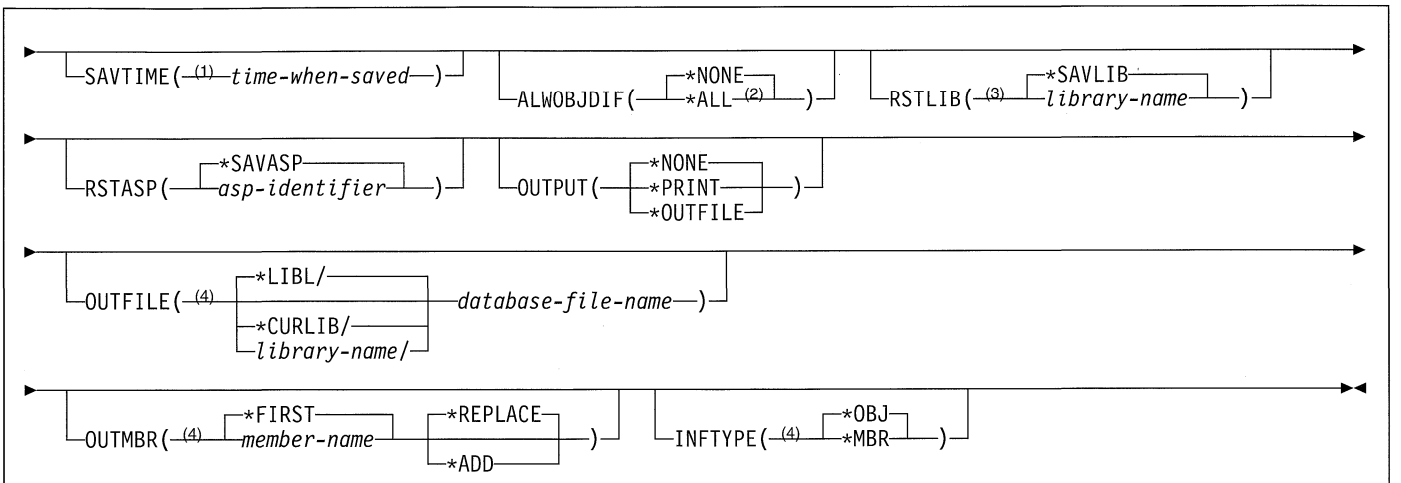
7 A maximum of 75 repetitions

8 Applies to tape devices only

9 If the FILEMBR parameter or ALWOBJDIF(*ALL) is specified, MBROPT(*MATCH) cannot be specified.

10 A maximum of 50 repetitions

P All parameters preceding this point can be specified in positional form.



Notes:

1 If VOL(*SAVVOL) is specified, SAVDATE and SAVTIME cannot be specified.

2 If the FILEMBR parameter or ALWOBJDIF(*ALL) is specified, MBROPT(*MATCH) cannot be specified.

3 If VOL(*SAVVOL) and RSTLIB(*SAVLIB) are specified, OPTION(*NEW) cannot be specified.

4 This parameter is valid only if OUTPUT(*OUTFILE) is specified.

Purpose

The Restore Object (RSTOBJ) command restores to the system a single object, or a group of objects, in a single library, that were saved on diskette, tape, or in a save file by using a single command. The types of objects that can be restored by this command are listed in the OBJTYPE parameter. They can be saved either as separate objects or as part of the library save operation. The RSTOBJ command restores the object description and contents of each object specified in the command.

For job queues, output queues, user-defined message queues, logical files, and data queues, only the object descriptions are restored; the contents of those objects are not restored. However, logical file access paths can be saved by a save command with ACCPTH(*YES) specified; if this is done, the access paths can be restored. More information on restoring access paths is in the *Database Guide*.

The command can be used to restore the objects even if the object storage was freed by the Save Object (SAVOBJ) command or Save Library (SAVLIB) command, or if the objects were deleted by the associated delete command for that object type. If the storage was not freed, each object is restored in the same area of storage that it previously occupied. If the version of the object being restored is larger than the version in the system (for example, data records that were deleted from the system still exist in the saved version of a file), the additional storage needed for the object is acquired. If the saved version of the object is smaller (for example, data records that are added to the system), the space that was acquired for the object remains assigned to that object and available for use by that object.

If logical file access paths were saved (ACCPTH(*YES) was specified when the objects were saved), the access paths are restored if (1) all based-on physical files are also being restored by the same restore command, (2) the logical file is also being restored by the same restore command, or the logical file already exists on the system (the same file exists, not a re-created version), and (3) MAINT(*IMMED or *DLY) is in effect for the logical file if it still exists on the system.

If the storage was freed, the system finds the storage space needed to store the contents (only the data portion) of each file, module, program, service program, journal receiver, and Structured Query Language (SQL) package. If the objects do not exist on the system because they were deleted or they are being restored in a different system, the system must find the storage to store everything (the description and the data portion) about each unknown object.

The user profile of the system default owner (QDFTOWN) becomes the default owner of objects restored in the system whose owner is not known to the system.

If an object already exists in the library to which it is restored, the public and private authorities of the existing object are kept. If the object does not exist in the library, all public authorities are restored, but any private authorities must be granted again. For an existing output queue object that is actively spooling during the restore operation, or a data queue that already exists in the library, none of the attributes are restored, and a diagnostic message is issued.

If an object is being restored over an existing object on the system, the object auditing value of the existing object is kept. If the object is being restored as new to the system, the object auditing value is restored from the media.

Restrictions:

1. The user must have use authority for the Create Save File (CRTSAVF) command when restoring a save file that does not currently exist on the system.
2. With the exception of overrides for the restore listing file and database files specified on the OUTFILE parameter, this command ignores all file overrides that are currently in effect for the job.
3. To use this command, the user must have the special authority *SAVSYS, specified in the user profile (SPCAUT parameter), or the following:
   - Read and add authority for, or be the owner of, the specified library.
   - Object existence authority for, or be the owner of, each object specified if the object already exists in the library on the system. Object existence authority and use authority are required for message queue objects. If the object does not already exist in the library on the system, the user must have add authority for the user profiles that own each object being restored.
   - If VOL(*SAVVOL) is specified, use authority to the saved-from library.
4. The RSTOBJ command does not restore the library's data dictionary or its associated database files. To do this, the RSTLIB command must be used.
5. The user must have use authority for the save file to restore from the save file. In addition, the user must have use authority for the device description when restoring from a tape or diskette.
6. If this command is used to restore a program, the copy of that program that is currently in the system must not be running while the program is being restored. If this occurs, the running program will not be restored.
7. Objects saved by separate commands must also be restored by separate commands. If a single command is used, some of the objects are not restored.
8. If the user domain object user space (*USRSPC), user index (*USRIDX), or user queue (*USRQ) is restored to a library that is not permitted in the system value QALWUSRDMN (allow user domain objects in libraries), the object is converted to the system domain.

Required Parameters

OBJ

Specifies the names of one or more objects to restore. The objects specified are restored from the first file that meets the search criteria and contains any of the objects. If a tape or diskette contains multiple files for the same library, specify the tape file sequence number or the diskette file label to ensure that all of the correct objects are restored.

If the file does not contain all of the objects specified, diagnostic messages are issued for the objects not found. The completion message contains a count of objects restored and objects not restored.

If the OBJTYPE parameter is not specified when the command is run, all of the object types listed in the description of the OBJTYPE parameter are restored, if they are in the specified library on the media or in the save file and have the specified names.

***ALL:** All the objects saved from the specified library are restored, depending on the values specified for the OBJTYPE and OPTION parameters.

generic-object-name:* Specify the generic name of the object. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk (*) substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix, for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

object-name: Specify one or more names of specific objects to restore. Both generic names and specific names can be specified in the same command.

SAVLIB

Specifies the name of the library from which the objects were saved. If RSTLIB is not specified, this is also the name of the library to which the objects are restored.

DEV

Specifies the name of the device used to restore the objects. The device name must already be known on the system by a device description.

***SAVF:** The restore operation is done using the save file specified on the save file (SAVF) parameter.

diskette-device-name: Specify the name of the diskette device used to restore objects.

tape-device-name: Specify the names of one or more tape devices used for the restore operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. Using more than one tape device permits one tape volume to be rewound and unloaded while another tape device processes the next tape volume.

Optional Parameters

OBJTYPE

Specifies the types of OS/400 system objects that are restored. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ALL:** All object types that are specified by name and that were saved from the specified library are restored. If *ALL is also specified for the OBJ parameter, then all objects saved for that library are restored.

object-type: Specify the value for each of the types of objects that are restored (such as command (*CMD), file (*FILE), or program (*PGM)). This value can be repeated; the maximum repetitions varies depending on the environment.

The object types that can be restored also can be saved and restored by the Save Library (SAVLIB), Save

Change Object (SAVCHGOBJ), Save Object (SAVOBJ), Restore Library (RSTLIB), and Restore Licensed Program (RSTLICPGM) commands. Data dictionaries (*DTADCT) are only restored with the RSTLIB command. Other object types can be saved only by the Save System (SAVSYS) command and restored by the OS/400 system restore operation, or saved by the Save Document Library Object (SAVDLO) command and restored by the Restore Document Library Object (RSTDLO) command.

Note: *CFGL object types can be restored using the RSTOBJ command only from objects saved on releases prior to V2R2M0. *CFGL object types saved on V2R2M0 and newer releases are restored using the Restore Configuration (RSTCFG) command.

VOL

Specifies the volume identifiers of the tape volumes or diskette volumes from which the objects are being restored. The volumes must be in the same order as they were when the library was saved. If the library was known to the system before the restore operation, the system can verify whether the volumes put on tape contain the current version of the library. The volume that contains the beginning of the file to be restored should be placed in the tape or diskette unit. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: The version of the objects that is restored is the first version found in the specified location, unless a specific version is identified by the SAVDATE and SAVTIME parameters.

***MOUNTED:** The objects are restored from the volumes placed in the device specified on the DEV parameter.

***SAVVOL:** The system, using the save/restore history information, determines which tape or diskette volumes contain the most recently saved version of the objects. When *SAVVOL is specified, the following operational characteristics and restrictions apply:

- If the characteristics of the device specified in the DEV parameter do not match the device and location of the most recently saved version of the library, an error message is sent to the *user*, and the function is ended.
- If the wrong volume is placed in a device in the location specified by the command, a message is sent to the *system operator* that identifies the first volume that must be placed in the device before the objects can be restored.
- When OBJ(*ALL) and OBJTYPE(*ALL) is specified, the objects are considered in the order they would appear in a display produced by the Display Library (DSPLIB) command. The object names and types specified in the RSTOBJ command are used to determine which file of saved objects is used in the restore operation. One file is produced for each SAVLIB or SAVOBJ command run. The file chosen is the one in which the first considered object was last saved. Objects that were not saved in the file chosen to be processed, or that were more recently saved, are not restored; an error message is sent to the user for each unrestored object.
- If *SAVVOL is specified, the SAVDATE and SAVTIME parameters cannot be specified.
- If *SAVVOL is specified and the RSTLIB value is equal to the SAVLIB value, OPTION(*NEW) cannot be specified.
- If *SAVVOL is specified, SEQNBR(*SEARCH) and LABEL(*SAVLIB) must be specified.

volume-identifier: Specify the identifiers of one or more volumes in the order in which they are placed in a device and used to restore the objects.

SEQNBR

Specifies, only when tape is used, the sequence number used for the restore operation.

***SEARCH:** The volume on a tape device is searched for a data file with an identifier that is a match for the SAVLIB parameter value; when a match is found, the object is restored. If the last operation on the device specifies ENDOPT(*LEAVE) (that is, the tape is positioned at the location where the last operation ended), the file search starts with the first data file beyond the current tape position. If ENDOPT(*LEAVE) was not used for the last operation (or if the tape was manually rewound after an ENDOPT(*LEAVE) operation), the search starts with the first data file on the volume.

sequence-number: Specify the sequence number of the file.

LABEL

Specifies the name that identifies the data file on the tape or diskette used for the restore operation. This label must have been specified on the save command.

***SAVLIB:** The file label is the name of the library specified on the SAVLIB parameter.

data-file-identifier: Specify the data file identifier (up to 17 characters) of the data file used for the restore operation. This option is valid only for a single library restore.

ENDOPT

Specifies, when tape is used, what positioning operation is automatically done on the tape volume after the restore operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

SAVF

Specifies the qualified name of the save file that contains the save data for the objects that are restored.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the save file from which to perform the restore.

OPTION

Specifies how to handle restoring each object.

***ALL:** All the objects in the saved library are restored to the library. All objects in the saved library are restored to the library on the system. Objects in the saved library replace the objects in the library on the system. Saved objects not on the system are added to the library on the system. Objects in the system library but not in the saved library remain in the library.

***NEW:** Only the objects in the saved library that do not exist in the current version of the library in the system are added to the library. Only objects not known to the library in the system are restored; known objects are not restored. This option adds objects that were deleted after they were saved or that are new to this library. If any saved objects have a current version in the library in the system, they are not restored; an informational message is sent for each object, but the restore operation continues.

***OLD:** Only objects in the library that have a saved version are restored; the current version of each object is replaced by the saved version. Only objects known to the library are restored. If any saved objects are no longer part of the current version of the library, they are not added to the library; an informational message is sent for each object, but the restore operation continues.

***FREE:** The saved objects are restored only if they exist in the library in the system with their space freed. The saved version of each object is restored in the system in its previously freed space. This option restores objects that had their space freed when they were saved. If any saved objects are no longer part of the current version of the library, or if the space is not free for any object, the object is not restored and an informational message is sent for each object. The restore operation continues and all of the freed objects are correctly restored.

FILEMBR

Specifies the database file members to restore. This parameter is made up of two parts: the file name and the member name. This parameter cannot be specified if MBROPT(*MATCH) is specified.

Notes:

1. Each database file specified on the FILEMBR parameter must also be specified on the OBJ parameter by either its complete name, a generic name, or *ALL.
2. The OBJTYPE parameter value must be *ALL, or it must include the *FILE value.
3. Generic names are not valid for the database file name, but are allowed for the member name.
4. Duplicate file names are not allowed.

Element 1: Database File Names

***ALL:** The member list following this value applies to all files specified on the OBJ parameter.

database-file-name: Specify the name of the database file that contains the members to be restored. Up to 50 of the file/member combinations can be specified for a single command.

Element 2: Member Names

***ALL:** All members are restored from the specified file.

***NONE:** No members are restored from the specified file. Only the file description is restored.

generic-member-name:* Specify the generic name of the members to be restored from the specified file. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix, for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

member-name: Specify the name of the member to be restored from the specified file.

Notes:

1. If the nongeneric member names are specified, all specified members must exist in the file in order for any part of the file to be restored.
2. If generic member names are specified, the file is not restored if it does not contain any members that match the generic names. For example, if the user specifies PAY* as a generic member name, and the system is unable to find any members whose names start with PAY, the file is not restored. If no files specified by the FILEMBR parameter are restored because no members with the specific generic name can be found, a diagnostic message is sent, the restore operation ends, and an escape

message is sent specifying the number of files not restored. If at least one of the files processed for the FILEMBR parameter is restored, the diagnostic message is not sent, and the number of files not restored is shown on the final completion message.

3. The members are first checked by using the MBROPT parameter. If a match is found, the members are checked against the list of the FILEMBR parameter.

MBROPT

Specifies, for database files currently on the system, which file members are restored.

If MBROPT(*MATCH) is used, the member list in the saved file must match, member for member, the current version on the system.

Note: Before restoring a file, the system verifies that the file and member creation dates of the known system objects match the creation dates of the objects to be restored. If this check fails, the file is not restored.

***MATCH:** The saved members are restored if the lists of the files where they reside match, member for member, the lists of the current system version.

***ALL:** All members in the saved file are restored.

***NEW:** Only new members (members not existing on the system) are restored.

***OLD:** Only members already existing on the system are restored.

SAVDATE

Specifies the date the objects were saved. If the most recently saved version is not the one being restored, or if multiple saved versions reside on the media volume, specify the date that identifies which version of the objects to restore. The date must be specified in the job date format. If separators, specified by the system value QDATSEP, are used, the value must be enclosed in apostrophes. If a volume identifier or VOL(*MOUNTED) is specified, but SAVDATE is not, the restored version of the objects is the first version found on the volume. This parameter is valid only if a volume identifier or VOL(*MOUNTED) is specified, or if a save file (SAVF parameter) is specified. Specify the date when the objects were saved.

SAVTIME

Specifies the time that the objects were saved.

The time can be specified with or without a time separator:

- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where hh = hours, mm = minutes, and ss = seconds.
- With a time separator, specify a string of 5 or 8 digits where the time separator specified for your job is used to separate the hours, minutes, and seconds. If you enter this command from the command line, the string must be enclosed in apostrophes. If a time separator other than the separator specified for your job is used, this command fails.

If a volume identifier or *MOUNTED is specified on the VOL parameter, but this parameter is not specified, the version of the objects to be restored is the first version found on the volume. This parameter is ignored when the SEQNBR parameter is specified.

This parameter is valid only if the SAVDATE parameter is also specified.

ALWOBJDIF

Specifies whether certain differences encountered during a restore operation are allowed. The differences include:

- Ownership: The owner of an object on the system is different than the owner of an object from the save operation.
- File creation date: The file creation date of the database file on the system does not match the creation date of the file that was saved.
- Member creation date: The file creation date of the database file on the system does not match the creation date of the member that was saved. The member creation date on the system does not match the creation date from the save operation.
- Validation value verification: The validation value created at the time an object was created does not match the validation value created during the restore operation of an object on a system with a QSECURITY level of 40 or higher.
- Authorization list linking: The object is being restored to a system different from the one on which it was saved.

Notes:

1. The user of this parameter must have *ALLOBJ special authority.
2. If *ALL is specified, *MATCH cannot be specified on the MBROPT parameter.

***NONE:** None of the differences described above are allowed on the restore operation. For validation value verification failure cases, the object is restored but ownership is transferred to QDFTOWN and all authorities are revoked. For authorization list cases, the object is restored, but the object is not linked to the authorization list, and public authority is set to *EXCLUDE. For all other cases, a diagnostic message is sent for the object, and the object is not restored.

***ALL:** All of the differences listed above are allowed for the restore operation. An informational message is sent, except for valid value verification and authorization list linking cases, and the object is restored. The following facts must be noted:

- If the owners of the object do not match, the object is restored, but the ownership and authorities it had before the restore operation are retained.
- Either the file itself or members of the file are restored based on the following facts:
  1. If there is a file level mismatch and ALWOBJDIF(*ALL) has been specified, the existing version of the file is renamed and the saved version of the file is restored.
  2. If there is a member level mismatch, the existing version of the member is renamed and the saved version of the member is restored.
- Validation value verification:
  1. For programs with no validation value stored on the media (in reference to program data saved on an OS/400 system prior to version 1, release 3.0), the program is restored and the system does not attempt to recreate it. Nothing is logged in the security journal and no informational messages are sent.
  2. For programs that do have a validation value stored on the media and the system fails to recreate the program, the original version of the program is restored and an entry is logged in the security journal and job log.
- The informational message causes a diagnostic message to be sent indicating that security or integrity changes occurred during the restore operation. This results in sending an escape message for the restore operation.
- If the user is restoring objects to a system different from the one on which they were saved and the objects are secured by an authorization list, specifying *ALL automatically links the objects to the authorization list again. If the authorization list does not exist on the new system, a message that includes the name of the missing list is issued and the public authority is set to *EXCLUDE.

RSTLIB

Specifies whether the objects are restored to a different library or to the same library where they were saved.

***SAVLIB:** The objects are restored to the same library from which they were saved.

library-name: Specify the name of the library to which the saved objects are restored.

RSTASP

Specifies whether the objects are restored to the same auxiliary storage pool (ASP) from which they were saved or to another specific ASP. This parameter affects only object types, journals, journal receivers, save files, and libraries. All other object types are restored to the ASP of the library unless the library is in a user ASP and the object type is not one of the valid user ASP object types. A list of the valid user ASP object types and additional information about user ASPs are in the *Basic Backup and Recovery Guide*.

Journals, journal receivers, and save files can be restored to a user ASP with the library in the system ASP if the objects do not already exist in a different ASP, and if the ASP to which they are being restored contains no libraries.

User ASPs that contain one or more libraries cannot contain valid ASP object types, journals, journal receivers, and save files in libraries on the system ASP.

***SAVASP:** The objects are restored to the same auxiliary storage pools from which they were saved.

asp-identifier: Specify the ASP identifier. When the specified ASP is 1, the objects are restored to the system ASP, and when the ASP is 2 through 16, the objects are restored to the user ASP specified.

Note: System libraries (QSYS, QUSRSYS, and QDOC) must not be created in or restored to user ASPs (2 through 16). Doing so can cause unpredictable results.

OUTPUT

Specifies whether a listing that shows information about the status of the object is created and directed to an output file. The listing shows the restore information and shows all objects, restored, not restored, and excluded. Information about each object's security is listed for the restored objects. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***NONE:** No output is created.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: You must specify the database file name on the OUTFILE parameter when OUTPUT(*OUTFILE) is specified.

OUTFILE

Specifies the qualified name of the database file to which the information about the object is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QASRRSTO in QSYS with the format name QSRRST as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which the output of the command is directed.

OUTMBR

Specifies the name of the database file member to which the output is directed when OUTPUT(*OUTFILE) is specified.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that receives the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it. If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

INFTYPE

Specifies the type of information directed to the database file. This parameter is valid only when OUTPUT(*OUTFILE) is specified.

***OBJ:** The list contains an entry for each object requested to be restored.

***MBR:** The list contains an entry for each object or, for database files, each member requested to be restored.

Examples

Example 1: Restoring Most Recently Saved Version

```
RSTOBJ    OBJ(PAYROLL)  SAVLIB(LIBX)  DEV(TAP01)
  OBJTYPE(*PGM) VOL(*SAVVOL)
```

This command restores to LIBX the program named PAYROLL that was saved from LIBX. The tape drive named TAP01 is used to restore the most recently saved version of the program.

Example 2: Specifying Save Date and Time

```
RSTOBJ    OBJ(PAY*)  SAVLIB(LIBX)  DEV(DKT01)  VOL(ABCD)
  OPTION(*OLD)  SAVDATE(102292)
  SAVTIME(143000)  RSTLIB(LIBY)
```

All objects whose names start with PAY and that were saved from the library named LIBX on diskette volume ABCD at 14:30:00 on 10/22/92 are restored to LIBY. Volume ABCD must be put on the diskette device named DKT01. Because OPTION(*OLD) is specified, the only objects restored are those having the same object name and type both in LIBY on the system and in LIBX on diskette.

Example 3: Adding a New Program to the QGPL Library

```
RSTOBJ    OBJ(NEWPROG)  SAVLIB(QGPL)  DEV(DKT01)
  OBJTYPE(*PGM)  VOL(PGMS)  OPTION(*NEW)
```

A new program named NEWPROG is added to the general purpose library, QGPL. It is restored from a volume labeled PGMS that is inserted in the diskette device named DKT01.

Example 4: Printing An Output List

```
RSTOBJ    OBJ(*ALL)  SAVLIB(LIB)  DEV(TAP01)  OBJTYPE(*PGM)
  VOL(SVOL)  SEQNBR(2)  SAVDATE(082392)
  SAVTIME(143000)  OUTPUT(*PRINT)
```

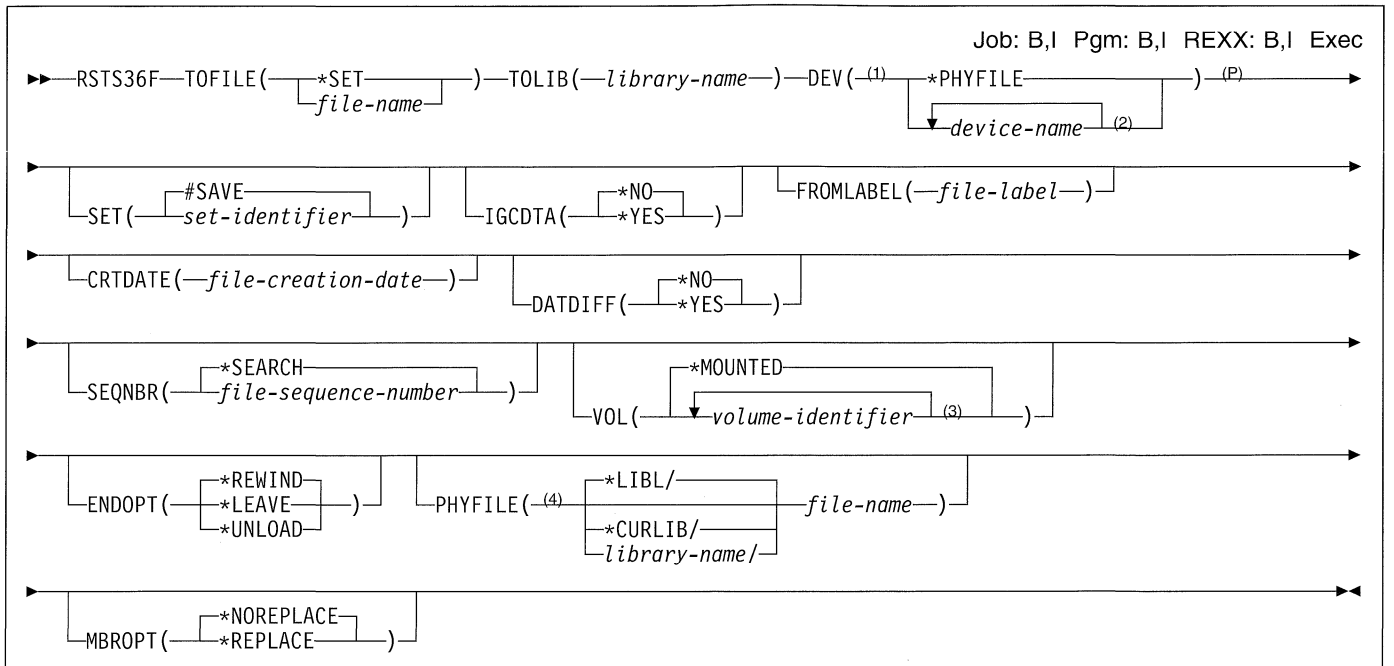
All programs that were saved from the library named LIB that exist in the saved version on the tape volume named SVOL, sequence number 2, saved at 14:30:00 on 08/23/92, are restored to the library named LIB. An output list of all objects restored as well as objects not restored is provided.

Example 5: Restoring Journal Receivers

```
RSTOBJ OBJ(*ALL) SAVLIB(BACKUP) DEV(*SAVF)
  OBJTYPE(*JRNRCV)  SAVF(SAVEJ) RSTASP(3)
```

All journal receivers that were saved from the library named BACKUP into the save file named SAVEJ are restored to the library named BACKUP. The journal receivers are restored to ASP 3 (unless they already exist in the library named BACKUP and are in a different ASP or ASP 3 contains a library).

RSTS36F (Restore System/36 File) Command



Notes:

1 Up to 4 devices may be specified if more than one tape device is being used. (Only one device is allowed with diskette.)

2 A maximum of 4 repetitions

P All parameters preceding this point can be specified in positional form.

3 A maximum of 50 repetitions

4 PHYFILE required if DEV(*PHYFILE) is specified.

Purpose

The Restore System/36 File (RSTS36F) command reads a file, creates a database physical or logical file (if it does not already exist), and copies any data into the file. This command restores to the system a single file or a group of files that are saved together using the System/36 save procedure. The input file can be a diskette file, tape file, or a database physical file on the AS/400 system. The file must have been created either on a System/36 using the SAVE system OCL procedure (or the equivalent OCL call of the $COPY SSP utility), or on the AS/400 system using the SAVS36F CL command.

The RSTS36F command accepts a diskette file created on a System/36, System/34, or System/32 using the $COPY utility. The RSTS36F command accepts tape files created on a System/36. System/34 or System/32 diskette offline multiple-volume files are not accepted by this command.

The RSTS36F command accepts a diskette file created as a compressed file.

If the file being restored does not exist in the library specified on the TOLIB parameter, it is created. A physical file member is added using the name syntax 'Myymmdd', which identifies the original creation date of the file. This naming convention is needed by the System/36 environment in order to support date-differentiated files.

If a file name is specified for the TOFILE parameter, the name must meet the AS/400 system naming standards. For more information about AS/400 system naming conventions, see Chapter 2, "Control Language Syntax."

If TOFILE(*SET) is specified, the files that are restored may have names that contain characters not allowed in an AS/400 simple object name. In this case, the file name is changed to an AS/400 system extended name and the file is restored.

If the name contains a blank, a single quotation mark, a double quotation mark, an asterisk, a question mark, or a device control character (hexadecimal 00 through 3F or hexadecimal FF), the invalid characters are replaced with underlines. The file is then restored using the resulting simple or extended name; for example, A*_? would become "A___"). If a file already exists with this new name, it is handled like any other name (see the MBROPT parameter).

If a file name is changed because of invalid characters, an informational message (CPF 2C1F) is sent to the recursion level above the program that is running this command. If the

name is changed from a simple name to an extended name, no message is sent.

If the restore function creates the file and the file was not previously secured, the new file is owned by the user issuing the RSTS36F command and the file is created with a default authority of *ALL (that is the same as AUT(*ALL)).

If the file was saved from the S/36 where the attributes were an extend value of zero or no value was specified, a default value of 32,767 divided by the record length is assigned. If an extend value of zero is required, use the Change Physical File (CHGPF) command (after the restore operation is completed) to set SIZE(*EXTEND) to zero. If the file was saved from the AS/400 system, the file is restored and the extend value does not change.

This function is intended only for exchanging files with a System/36.

Restrictions:

1. The following authorities are required when running on a system using resource security:
   - *USE, *SECADM, and *ALLOBJ authorities for this command
   - *USE authority for the library specified in the TOLIB parameter
   - *CHANGE authority for the library specified in the TOLIB parameter when restoring a file that does not already exist on the system
   - *CHANGE and *OBJMGMT authority for the existing file (needed to add a new member) when restoring a date-differentiated physical file and a file by the same name but with a different creation date
   - *ALL authority for the file when restoring to an existing physical file with the same creation date and MBROPT(*REPLACE) specified
   - *CHANGE authority for the based-on physical file (this physical file was referred to as the *parent file* on System/36) if the file being restored is a System/36 alternative index file (that is, a logical file)
   - *USE authority for the diskette device description object and *USE authority for device file QSYSDKT in library QSYS, when restoring from diskette
   - *USE authority for the tape device description object and *USE authority for device file QSYSTAP in library QSYS, when restoring from tape
   - *USE authority for the file and *USE authority for the library that contains the file (PHYFILE parameter) if restoring from a database physical file
   - If the file doesn't exist on the system but a file authorization holder object by the same name does exist, *ALL authority or ownership for the authorization holder object
2. There is no replace function supported when restoring System/36 alternative index files (logical files). If restoring an alternative index file, no file object by the same name can already exist in the specified library.
3. If restoring a logical file, the based-on physical file must already exist in the library specified for the TOLIB parameter.
4. The AS/400 system files that are the same as the System/36 date-differentiated files are multiple-member physical files. Date-differentiated alternative index files are not supported. The data for a physical file is stored in a member that is named using the syntax 'Myymmdd' where 'yymmdd' represents the original creation date (in year/month/day format) of the file.

   Because all members of a physical file share the same file attributes (for example, record length and key information), date-differentiated files with the same name that are restored to the same library are required to have the same file attributes. If an attribute mismatch is present, the files are not restored.
5. Object-level and record-level functions, including read operations, should not be used for a file being restored by the RSTS36F command. If another operation is being done at the same time on the file (for example, moving the file or reading or adding records), the restore operation stops if it cannot allocate the file.

Required Parameters

TOFILE

Specifies the name of a single file that is restored to the system or that a group of files that are all saved at the same time using the System/36 SAVE procedure (a Save All Set) are to be restored to the system.

***SET:** A group of files from a Save All Set are restored to the system.

file-name: Specify the file name given to a single file when the file is restored to the system.

TOLIB

Specifies the library that will contain the file being restored from the input file.

DEV

Specifies the names of the devices used for the restore operation. Each device name must already be known on the system by a device description.

***PHYFILE:** The input file is the database physical file specified by the PHYFILE parameter.

device-name: Specify the name of the diskette device or the names of tape devices that are used for the operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. When more than one tape volume is used, using more than one tape device permits one tape volume to be rewound or unloaded while another tape device processes the next tape volume. More information on device names is in the *APPC Programmer's Guide*.

Optional Parameters

SET

Specifies the name used to identify the Save All Set files saved on the diskette or tape by the SAVE procedure or the $COPY utility on the System/36, System/34, or System/32.

#SAVE: The files are restored from a Save All Set with a set identifier of #SAVE.

set-identifier: Specify the set identifier of the Save All Set that is restored. Valid values range from 1 through 8 characters. The first character must be alphabetic (A through Z, #, $, or @). The remaining characters can be any combination of characters (numeric, alphabetic, and special) except a comma (,), an apostrophe ('), or a blank.

IGCDTA

Specifies whether the file being restored can contain double-byte character set (DBCS) data.

***NO:** The file being restored cannot contain DBCS data. If a file already exists by the name specified on the TOFILE parameter in the specified library and the file allows DBCS data, an informational message is sent and the restore operation continues. The resulting file allows DBCS data (the file's IGC attributes are not changed by the restore operation).

***YES:** The file processes DBCS data.

FROMLABEL

Specifies the label of the diskette or tape file that contains the file being copied to the AS/400 system (it identifies the input file for the restore operation). Up to 8 characters can be used to label the diskette or tape file.

For a group restore operation, this parameter can be used to specify the diskette or tape file label with a save all set where the restore operation is to begin. If no value is specified, the restore operation begins with the first file in the set.

If no value is specified, the value of the TOFILE parameter is used as the diskette or tape label. If the TOFILE parameter is an extended name, the characters between the quotation marks are used as the label. For example, specifying TOFILE("MIKE&BOB") and no FROMLABEL parameter would be the same as specifying FROMLABEL('MIKE&BOB').

CRTDATE

Specifies the creation date of the object.

The date must be entered in the job date format (DATFMT); if separators, specified by the job date separator character (DATSEP), are used, the value must be enclosed in apostrophes. The CRTDATE parameter cannot be specified if the FROMLABEL parameter is not specified.

For diskettes coming from a System/36, there may be multiple files on one diskette with the same name and different creation dates. Specifying a value for the CRTDATE parameter is the only way of choosing which diskette file to use.

For tapes from System/36 or AS/400 system, there can be multiple files on a tape with the same name. The user can use the CRTDATE parameter or SEQNBR parameter to choose the correct file. If a numeric SEQNBR parameter and the CRTDATE parameter are specified, the SEQNBR parameter takes precedence (if the file at the specified sequence number has a different creation date, the restore operation ends).

Specify the creation date of the tape or diskette file used for the restore operation. The date specified is changed to Julian format (cyyddd) for tape files and international format (yymmdd) for diskette files.

If no value is specified, the diskette file or tape file creation date is not used to select the input file. For diskette, take the first file in the diskette volume table of contents (VTOC) with the name specified on the FROMLABEL parameter. For tape, take the first file on the tape with the name specified on the FROMLABEL parameter. If the previous tape operation had specified ENDOPT(*LEAVE), the search for a matching file does not start at the beginning of the tape reel.

DATDIFF

Specifies whether the restore operation should allow date-differentiated files (that is, multiple files with the same name but different file creation dates).

***NO:** Date-differentiated files are not allowed.

If a file already exists by the name specified on the TOFILE parameter in the specified library, and the file contains a member with a different creation date (as determined by the member name), an error message is sent and the file is not restored.

If a file already exists by the name specified on the TOFILE parameter in the specified library, and the file contains a member with the same creation date (as determined by the member name), either the data in that member is replaced (if MBROPT(*REPLACE) is specified) or an error message is sent (if MBROPT(*NOREPLACE) is specified).

If no file exists by the name specified on the TOFILE parameter in the specified library, the file is restored normally.

***YES:** Date-differentiated files are allowed.

If a file already exists by the name specified on the TOFILE parameter in the specified library, and the file contains a member with a different creation date (as determined by the member name), a member is added to the file (only if the file attributes match), and the file's data is restored into the new member.

If a file already exists by the name specified on the TOFILE parameter in the specified library, and the file contains a member with the same creation date (as determined by the member name), either the data in that member is replaced (if MBROPT(*REPLACE) is specified) or an error message is sent (if MBROPT(*NOREPLACE) is specified).

If no file exists by the name specified on the TOFILE parameter in the specified library, the file is restored normally.

SEQNBR

Specifies, only when tape is used, which sequence number is used for the restore operation.

***SEARCH:** The volume in the device is searched for a data file with an identifier that matches the FROMLABEL parameter value; when a match is found, the data file is restored. If the last operation on the tape device specified ENDOPT(*LEAVE), the file search starts with the data file at the current tape position. Otherwise, the search starts with the first data file on the tape volume. The header label name from the file found by the SEQNBR parameter is compared to the name specified for the FROMLABEL parameter. If the names are the same, the restore operation can continue. If the names are different, the restore operation ends.

file-sequence-number: Specify the sequence number of the file that is used.

VOL

Specifies the volume identifiers of the tape volumes or diskette volumes from which the objects are being restored. The volumes must be in the same order as they were when the library was saved. If the library was known to the system before the restore operation, the system can verify whether the volumes put on tape contain the current version of the library. The volume that contains the beginning of the file to be restored should be placed in the tape or diskette unit. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The objects are restored from the volumes placed in the device specified on the DEV parameter.

volume-identifier: Specify the volume identifiers of the tapes or diskettes to be used for restoring the file. For each tape or diskette volume name, up to 6 characters can be specified using any combination of letters and digits. Up to 50 volume names can be specified.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

Note: This parameter is ignored if a diskette device is specified in the DEV parameter.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

PHYFILE

Specifies the qualified name of the database physical file that is used as the input file for the restore operation.

The file must have a record length of 256 bytes.

If the specified file does not exist or is not a physical file, the file is not restored. If the physical file contains multiple members, the first member of the file is used.

The name of the physical file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the database file that is used as the input file for the restore operation.

MBROPT

Specifies whether the new records replace or are added to the existing records.

***NOREPLACE:** If a file already exists by the name specified on the TOFILE parameter in the specified library, and the file contains a member with the same creation date (as determined by the member name), an error message is sent and the data in that member is not replaced.

***REPLACE:** If a file already exists by the name specified on the TOFILE parameter in the specified library, and the file contains a member with the same creation date as the file being restored, the data in that member of the file is replaced.

Examples

Example 1: Restoring From Diskette

```
RSTS36F  TOFILE(ACCTRCV)  TOLIB(QS36F)  DEV(I1)
  CRTDATE('01/22/85')  VOL(SAVE1)
```

This command restores the file ACCTRCV into library QS36F. Assuming that I1 is the name of a diskette device description object, the file is restored from the diskette placed in the diskette device. The diskette must have a volume name of SAVE1. The diskette file used for the restore must have a file label of ACCTRCV and a creation date of January 22, 1985 (assuming the job date format is *MDY and the date separator is a '/').

Example 2: Restoring From Tape

```
RSTS36F  TOFILE(PAY.VIEW)  TOLIB(PAYLIB)  DEV(T1)
  FROMLABEL('P*V')  ENDOPT(*LEAVE)
```

The file P*V is restored from device T1 as a file named PAY.VIEW in library PAYLIB. Assuming T1 is a tape device, the file is copied from one or more tapes that are on device

T1. No check is made on the tape volume name. When the restore operation ends, the tape is left positioned at the end of tape file P*V.

Example 3: Restoring from a Physical File

```
RSTS36F  TOFILE(ACCTPAY)  TOLIB(QS36F)  DEV(*PHYFILE)
  PHYFILE(NETLIB/SENDFILE)
```

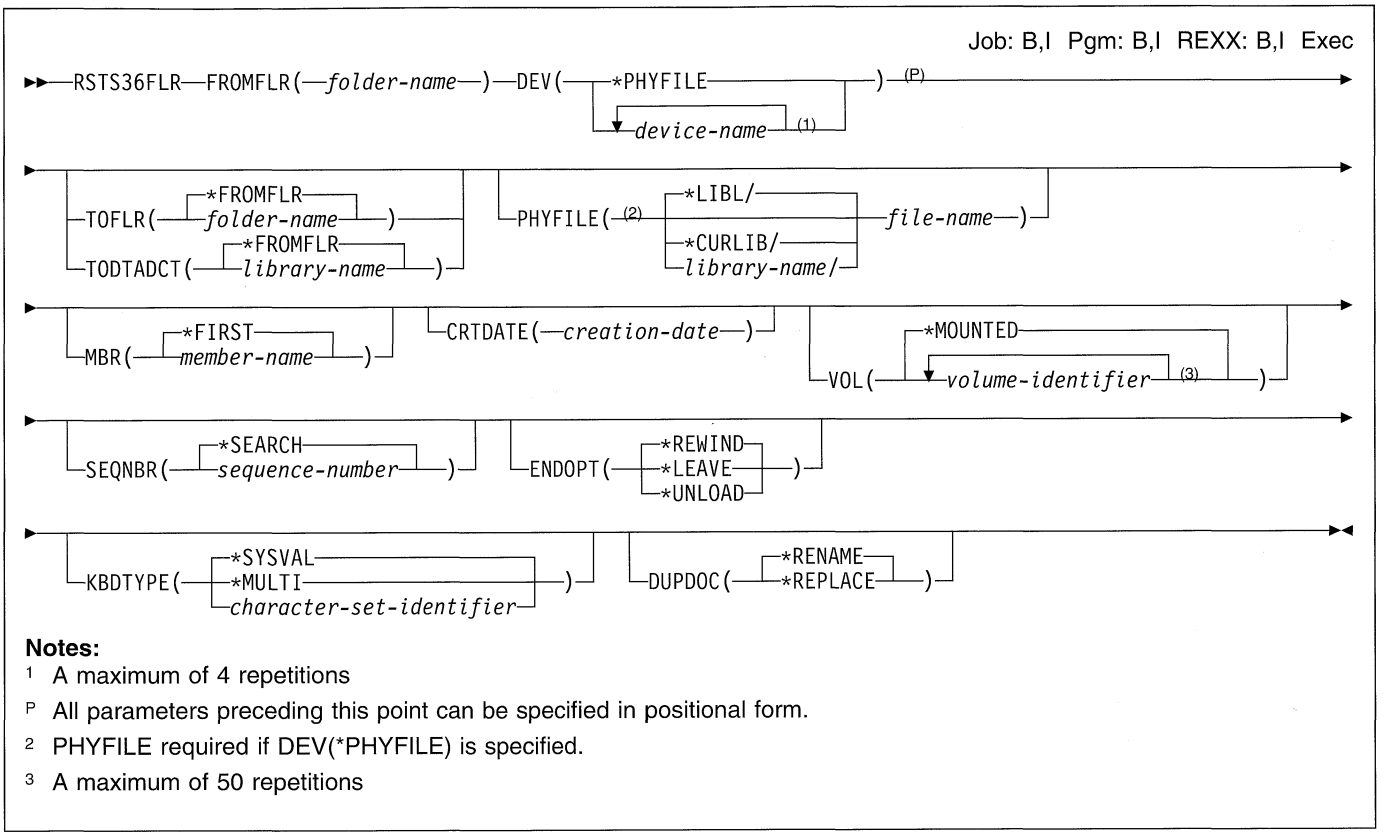
This command restores the file ACCTPAY in library QS36F from physical file SENDFILE in library NETLIB.

Example 4: Specifying Sequence Numbers

```
RSTS36F  TOFILE(*SET)  TOLIB(QS36F)  DEV(T1 T2)
  SET(PAYFILES)  FROMLABEL(FILE10)  MBROPT(*REPLACE)
  DATDIFF(*YES)  SEQNBR(*SEARCH)  VOL(*MOUNTED)
  ENDOPT(*REWIND)
```

This command restores a subset of the files in the save all set called PAYFILES to library QS36F from tape. The restore operation begins with a tape file whose label is file 10. If one of the files being restored already exists in library QS36F with the same creation date as the saved file, the file is replaced. If a file already exists in library QS36F with a different creation date, a new date-differentiated number is added to the file. The restore operation uses the tape volumes that are placed in tape drives T1 and T2. After the restore operation is complete, the last tape volume is rewound to the beginning of the tape.

RSTS36FLR (Restore System/36 Folder) Command



Notes:

1 A maximum of 4 repetitions

P All parameters preceding this point can be specified in positional form.

2 PHYFILE required if DEV(*PHYFILE) is specified.

3 A maximum of 50 repetitions

Purpose

The Restore System/36 Folder (RSTS36FLR) command reads a diskette, tape, or database file that was prepared by the System/36 SAVEFLDR command. The data is converted so that it can be used by the AS/400 system. Both compressed and uncompressed data is supported when restoring data from a diskette.

Two types of System/36 folders are supported as input: document folders and interactive data definition utility (IDDU) folders.

- Document folders are converted to AS/400 system folders, and the documents contained in them are converted to AS/400 system documents. If a System/36 document is in the process of being edited when the SAVEFLDR is done, there are two existing versions of the document: an unchanged original, and one containing modifications. When this folder is restored on the AS/400 system, the unchanged original and the modification version are created. If the editing is completed by using OfficeVision/400 on the AS/400 system, the modification version is removed.
- System/36 IDDU folders do not create folders on the AS/400 system. Instead, a library is created and the data dictionary is converted to an AS/400 system data dictionary.

Notes:

1. When restoring a folder from the System/36, the user that issues the restore command becomes the owner and the one authorized to the folder.
2. When restoring a folder from the System/36 that contains an invalid name for the AS/400 system, the name is automatically changed to a different name generated by the system.

Restrictions:

1. The user must have *USE authority to the from-folder, *CHG authority to the to-folder, and *USE authority to the device file or device description.
2. If the source folder is a document folder, the user must be enrolled in the system distribution directory and have either operational and data authorities to the folder or *ALLOBJ special authority. If replacing a document, the user must have *ALL authority to the document.

Required Parameters

FROMFLR

Specifies the name of the folder that is being restored. Specify the name of the folder.

DEV

Specifies the name of the device from which to restore the specified folder. The device name must be known to the system by a device description.

***PHYFILE:** The folder is in a physical database file. If DEV(*PHYFILE) is specified, then the PHYFILE parameter must be specified.

device-name: Specify the name of the diskette device or the names of tape devices that are used for the operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. When more than one tape volume is used, using more than one tape device permits one tape volume to be rewound or unloaded while another tape device processes the next tape volume. More information on device names is in the *APPC Programmer's Guide*.

Optional Parameters

TOFLR

Specifies the name of the folder in which to put the restored folder's contents. This parameter is used only for document folders. An error message is sent if this parameter is specified for any other type of folder.

***FROMFLR:** The old name of the folder specified in the FROMFLR parameter is used for the name of the restored folder.

folder-name: Specify the new name for the folder being restored.

TODTADCT

Specifies the name of the library where an IDDU dictionary (folder) is restored. This parameter is used only for IDDU dictionaries. An error message is sent if this parameter is specified for any other type of folder. When this parameter is specified, a library is created and a data dictionary is created in the new library. An error message is sent if a library with the specified name already exists.

***FROMFLR:** The old name of the folder specified in the FROMFLR parameter is used for the name of the restored library.

library-name: Specify the name of the library where the restored folder is placed.

PHYFILE

Specifies the qualified name of the database file that contains the System/36 folder.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the database file that contains the System/36 folder.

MBR

Specifies the name of the file member being restored.

***FIRST:** The first member in the database file is used.

member-name: Specify the name of the file member that contains the restored System/36 folder.

CRTDATE

Specifies the creation date of the object.

The date must be entered in the job date format (DATFMT); if separators, specified by the job date separator character (DATSEP), are used, the value is enclosed in apostrophes.

Diskettes from a System/36 may contain several files with identical names and different creation dates. Specifying a value for the CRTDATE parameter is the only way to choose which diskette file to use.

Tapes from a System/36 or AS/400 system may contain several files with identical names and different creation dates. The user can use the CRTDATE or SEQNBR parameter to choose the correct file. If both a numeric SEQNBR parameter and the CRTDATE parameter are specified, the SEQNBR parameter takes precedence. If the file at the specified sequence number has a different creation date, the restore operation ends.

Specify the creation date of the tape or diskette file used for the restore operations. The date specified is changed to Julian format (cyyddd) for tape files and international format (yymmdd) for diskette files.

VOL

Specifies the volume identifiers of the tape volumes or diskette volumes from which the objects are being restored. The volumes must be in the same order as they were when the library was saved. If the library was known to the system before the restore operation, the system can verify whether the volumes put on tape contain the current version of the library. The volume that contains the beginning of the file to be restored should be placed in the tape or diskette unit. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The objects are restored from the volumes placed in the device specified on the DEV parameter.

volume-identifier: Specify the identifier of one or more volumes in the order in which they are used. Up to 6 characters identify each volume.

SEQNBR

Specifies the sequence number of the file being restored.

***SEARCH:** The tape is searched for the first sequence number that contains the specified folder.

sequence-number: Specify the sequence number of the file.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

Note: This parameter is ignored if a diskette device is specified in the DEV parameter.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

KBDTYPE

Specifies the character set and code page of the System/36 that created the data in the folder.

***SYSVAL:** The System/36, from which the folder was saved, uses the same character set and code page as the AS/400 system.

***MULTI:** The System/36, from which the folder was saved, is configured as a multinational system.

character-set-identifier: Specify one of the identifiers from Table 28 if the System/36 is a non-multinational system and the AS/400 system is not using the same character set and code page as the System/36. On the System/36, a list of language or country names is presented from which to make a selection. If multinational is chosen, the System/36 maps it into a single code regardless of language. For example, if Denmark is chosen and not multinational, use DMB with RSTS36FLR. If Denmark is chosen with multinational on the System/36, use *MULTI with RSTS36FLR.

Table 70. Keyboard Mapping Table

Language/Country	Identifier	ASCII Device Group
Arabic X/Basic	CLB	D
Austria/Germany	AGB	A, B
Austria/Germany Multinational	AGI	A, B
Belgium Multinational	BLI	B
Brazilian Portuguese	BRB	
Canadian French	CAB	A, B
Canadian French Multinational	CAI	A, B
Cyrillic	CYB	
Denmark	DMB	B
Denmark Multinational	DMI	B
Finland/Sweden	FNB	B
Finland/Sweden Multinational	FNI	B
France (Azerty)	FAB	A, B
France (Azerty) Multinational	FAI	A, B
France (Qwerty)	FQB	
France (Qwerty) Multinational	FQI	
Greece	GNB	
Greece	GKB	
Hebrew	NCB	D
Iceland	ICB	
Iceland Multinational	ICI	
International	INB	
International Multinational	INI	
Italy	ITB	A, B
Italy Multinational	ITI	A, B
Japan English	JEB	
Japan English Multinational	JEI	
Japan Kanji (For PS*/55 and 5295 display stations)	JKB	
Japan United States Basic	JUB	
Japan Katakana (For 5251, 5291, 5292, and 3180 Katakana display stations)	KAB	
Korea	KOB	
Latin-2/ROECE	ROB	
Netherlands	NEB	
Netherlands Multinational	NEI	
Norway	NWB	B
Norway Multinational	NWI	B
Portugal	PRB	B
Portugal Multinational	PRI	B
Simplified Chinese	RCB	
Spain	SPB	B
Spain Multinational	SPI	B
Spanish Speaking	SSB	B
Spanish Speaking Multinational	SSI	B
Sweden	SWB	B
Sweden Multinational	SWI	B
Switzerland/French Multinational	SFI	B
Switzerland/German Multinational	SGI	B
Thai	THB	
Traditional Chinese	TAB	
Turkey	TKB	
United Kingdom	UKB	A, B
United Kingdom Multinational	UKI	A, B
United States/Canada	USB	A, B, C
United States/Canada Multinational	USI	A, B, C
Languages of the former Yugoslavia	YGI	

DUPDOC

Specifies whether existing documents are renamed or replaced. This parameter is ignored when restoring a data dictionary.

***RENAME:** If a folder already exists with the name specified in the TOFLR parameter and a document in that folder exists with the same name as an incoming document, the incoming document is created in the folder with a new system-generated name. All other documents are created in the specified folder.

***REPLACE:** If a folder already exists by the name specified in the TOFLR parameter, and a document in

that folder exists with the same name as an incoming document, the existing document is replaced. All other documents are created in the specified folder.

Examples

Example 1: Restoring From Diskette

```
RSTS36FLR  FROMFLR(TXTDRA)  DEV(DKT01)
  CRTDATE('01/16/88')  VOL(SAVE1)
```

This command restores the folder TXTDRA. Assuming that DKT01 is the name of a diskette device description, the folder is restored from the diskette placed in the diskette device. The diskette must have a volume name of SAVE1 and the folder must have a creation date of January 16, 1988.

Example 2: Restoring From Tape

```
RSTS36FLR  FROMFLR(TXTMEL)  DEV(TAP1)
  TOFLR(TXTNEW)  SEQNBR(7)
  ENDOPT(*UNLOAD)
```

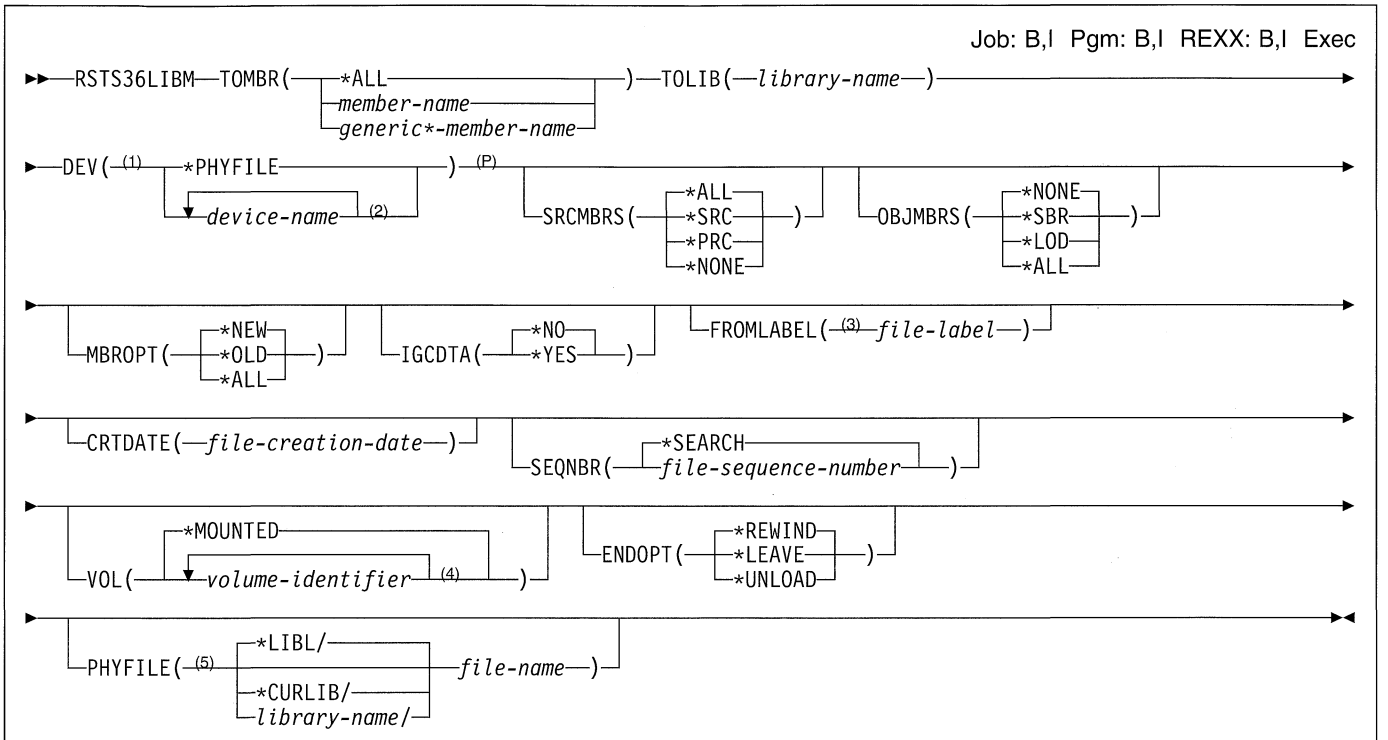
This command restores the folder TXTMEL to a folder named TXTNEW. Assuming TAP1 is a tape device, the folder is restored from one or more tapes that are on device TAP1. The folder must have a sequence number of 7. The tape is unloaded from the device at the end of processing.

Example 3: Restoring to a Data Dictionary

```
RSTS36FLR  FROMFLR(DONDCT)  DEV(*PHYFILE)
  TODTADCT(NEWDCT)  PHYFILE(MYLIB/MYFILE)
  MBR(THISONE)
```

This command restores the folder DONDCT to a data dictionary named NEWDCT. Assuming that MYFILE is a physical file located in library MYLIB and contains the member THISONE, the folder is restored from member THISONE.

RSTS36LIBM (Restore System/36 Library Members) Command



Notes:

1 Up to four devices may be specified if more than one tape device is being used. (Only one device is allowed with diskette.)

2 A maximum of 4 repetitions

P All parameters preceding this point can be specified in positional form.

3 FROMLABEL required if tape or diskette devices on DEV parmeter.

4 A maximum of 50 repetitions

5 PHYFILE required if DEV(PHYFILE) is specified.

Purpose

The Restore System/36 Library Members (RSTS36LIBM) command reads a file containing library members, creates database source or data file members on the AS/400 system, and copies the member data from the file into each restored member. The input file can be a diskette file, tape file, or database physical file on the AS/400 system. The file could have been created on a System/36, a System/34, or a System/32 using either the FROMLIBR or SAVELIBR system OCL procedure (or the equivalent call of $MAINT), or on an AS/400 system using the SAVS36LIBM CL command.

Diskette files created on a System/34 using the BACKUP procedure or $BACK utility are not accepted by this command. Compressed SAVELIBR diskette files (used by IBM to distribute system libraries for System/36 after release 5.0) are not accepted by this command.

In System/36 terms, the input file format could be a SAVELIBR diskette file or tape file, a record-mode LIBRFILE diskette file, tape file, or physical file, or a sector-mode LIBRFILE diskette file, tape file, or physical file. In other words, the input file can be any diskette file, tape file, or physical file created by the System/36 $MAINT SSP utility.

If the library identified by the TOLIB parameter value does not exist, it is created. Also, the source files QS36SRC and QS36PRC are created if they do not exist in the restore-to library.

If the restore operation creates the library, the new library is owned by the user running the RSTS36LIBM command and the library is created with a default authority of *ALL (that is, the same as *AUT(*ALL)*).

If a sector-mode FROMLIBR file or a SAVELIBR file created on a System/36 is being restored, data files QS36LOD and QS36SBR may also be created to hold restored load and subroutine members. Restored load and subroutine members are not converted on the current system.

Restrictions:

- The following authorities are required when running on a system using resource security:
  1. *SECADM and *ALLOBJ authorities
  2. *USE authority for this command and *USE authority for the Create Library (CRTLIB) command if the library needs to be created
  3. *USE authority for the CRTSRCPF command if QS36SRC or QS36PRC must be created
  4. *USE authority for the CRTPF command if QS36LOD or QS36SBR must be created
  5. *CHANGE authority for the library specified in the TOLIB parameter
  6. *CHANGE and *OBJMGMT authority for file QS36SRC in the specified library if restoring source library members
  7. *CHANGE and *OBJMGMT authority for file QS36PRC in the specified library if restoring procedure library members
  8. *CHANGE and *OBJMGMT authority for file QS36LOD in the specified library if restoring load library members
  9. *CHANGE and *OBJMGMT authority for file QS36SBR in the specified library if restoring subroutine library members
  10. *USE authority for the diskette device description object, *USE authority for device file QSYSDKT in library QSYS if restoring from diskette
  11. *USE authority for the tape device description object and *USE authority for device file QSYSTAP in library QSYS, if restoring from tape
  12. *USE authority for the file and *USE authority for the library that contains the file (PHYFILE parameter) if restoring from a database physical file
- No object-level or record-level operations should be active for files QS36SRC, QS36PRC, QS36SBR, and QS36LOD while members are being restored by RSTS36LIBM. If the necessary files cannot be allocated exclusively, no members are restored.
- The member name or generic member name specified (TOMBR parameter) must meet AS/400 system naming standards.

  If a generic member name or *ALL is specified, a member may be selected to be restored that has a name containing characters not allowed in an AS/400 system *simple* object name. In this case, the member name is restored using the AS/400 system *extended* name syntax (for example, A!B would become *'A!B'*).

If the name contains a blank, a single quotation mark (') a double quotation mark ("), an asterisk (*), a question mark (?), or a device control character (hexadecimal '00'-'3F' or hexadecimal 'FF'), these characters are replaced by underlines and the member is restored using the resulting simple or extended name (for example, A*/? would become *A_/_* and A? would become *A_*).

An informational message is sent each time invalid characters are replaced to get a valid name. An additional informational message is sent if the resulting name change caused a member to be replaced. No message is sent if a member is restored using the extended name syntax without replacing invalid characters.

Required Parameters

TOMBR

Specifies the names of the members to restore.

***ALL:** All members of the specified member types are restored.

member-name: The members that have the specified member name are restored.

generic-member-name:* All members that have the specified generic member name are restored. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be restored only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

TOLIB

Specifies the library that will contain the members to restore from the input file.

DEV

Specifies the names of the devices to use for the restore operation. Each device name must already be known on the system by a device description.

***PHYFILE:** The input file is the database physical file specified by the PHYFILE parameter.

device-name: Specify the name of the diskette device or the names of tape devices that are used for the operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. When more than one tape volume is used, using more than one tape device permits one tape volume to be rewound or unloaded while another tape device processes the next tape volume. More information on device names is in the *APPC Programmer's Guide*.

Optional Parameters

SRCMBRS

Specifies which source member types (source and procedure members on System/36) are restored.

***ALL:** All source and procedure members in the input file that match the member name specified (TOMBR parameter) are restored. For example, if TOMBR(PAY*) and SRCMBRS(*ALL) is specified, all source and proce-

dure members that start with the letters 'PAY' are restored.

***SRC:** Only System/36 source members (to file QS36SRC) that match the member name specified (TOMBR parameter) are restored.

***PRC:** Only System/36 OCL procedure members (to file QS36PRC) that match the member name specified (TOMBR parameter) are restored.

***NONE:** No source or procedure library members are restored.

OBJMBRS

Specifies which object member types (load and subroutine members on System/36) are restored. Because the System/36 and the AS/400 system are not object compatible, any restored members are not immediately useable after the restore operation. IBM-supplied commands or user-written operations must be run to convert the object member to a useable AS/400 system object.

***NONE:** No System/36 object members are restored.

***SBR:** Only System/36 subroutine members (to file QS36SBR), which match the member name specified (TOMBR parameter), are restored.

***LOD:** Only System/36 load members (to file QS36LOD), which match the member name specified (TOMBR parameter), are restored.

***ALL:** All load and subroutine members in the input file which match the member name specified (TOMBR parameter), are restored. For example, if TOMBR(CONF1) and OBJMBRS(*ALL) are specified, load and subroutine members with the member name 'CONF1' are restored.

MBROPT

Specifies, for database files currently on the system, which file members are restored.

***NEW:** Only new members (members that do not already exist in the appropriate source or data file) are restored.

***OLD:** Only old members (members that already exist in the appropriate file) are restored. The existing member is replaced by the copy of the member restored from the file.

***ALL:** All members are restored, even if they already exist in an AS/400 system source or data file. Members that do not already exist are created, and members that do exist are replaced.

IGCDTA

Specifies whether the source and procedure members being restored can contain double-byte character set (DBCS) data.

Note: This attribute is used if the restore operation needs to create source files QS36SRC and QS36PRC to hold the restored library members. If the QS36SRC or QS36PRC source file already exists in the library specified on the TOLIB parameter and the file's DBCS capability does not match the parameter, an error message is sent and no member is restored.

| ***NO:** The file does not process DBCS data.

| ***YES:** The file processes DBCS data.

FROMLABEL

Specifies the label (8 characters maximum) of the diskette file or tape file that contains the members to copy to the AS/400 system (it identifies the input file for the restore operation). If the DEV parameter value is not *PHYFILE, a FROMLABEL value must be specified.

CRTDATE

Specifies the creation date of the object.

The date must be entered in the job date format (DATFMT); if separators are used, specified by the job date separator character (DATSEP), the value must be enclosed in apostrophes.

For diskettes coming from a System/36, there may be several files on one diskette with the same name and different creation dates. Specifying a value for the CRTDATE parameter is the only way of choosing which diskette file to use.

For tapes from System/36 or the AS/400 system, there can be several files on a tape with the same name. The user can use the CRTDATE or SEQNBR parameter to choose the correct file. If a numeric SEQNBR parameter and the CRTDATE parameter are specified, the SEQNBR parameter takes precedence. If the file at the specified sequence number has a different creation date, the restore operation ends.

Specify the creation date of the tape file or diskette file to use for the restore operation. The date specified is changed to Julian format (cyyddd) for tape files and international format (yymmdd) for diskette files.

If no value is specified, the diskette file or tape file creation date is not used to select the input file. For diskette files, take the first file in the diskette VTOC with the name specified on the FROMLABEL parameter. For tape files, take the first file on the tape with the name specified on the FROMLABEL parameter. If the previous tape operation had specified ENDOPT(*LEAVE), the search for a matching file does not start at the beginning of the tape reel.

SEQNBR

Specifies, only when tape is used, which sequence number is used for the restore operation.

***SEARCH:** The volume that is placed in the device is searched for a data file with an identifier that matches the FROMLABEL parameter value; when a match is found, the data file is restored. If the last operation on the tape device specified ENDOPT(*LEAVE), the file search begins with the data file at the current tape position. Otherwise, the search begins with the first data file on the tape volume. The header label name from the

file found by the SEQNBR parameter is compared to the name specified for the FROMLABEL parameter. If the names are the same, the restore operation can continue. If the names are different, the restore operation ends.

file-sequence-number: Specify the sequence number of the file that is used.

VOL

Specifies the volume identifiers of the tape volumes or diskette volumes from which the objects are being restored. The volumes must be in the same order as they were when the library was saved. If the library was known to the system before the restore operation, the system can verify whether the volumes put on tape contain the current version of the library. The volume that contains the beginning of the file to be restored should be placed in the tape or diskette unit. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The objects are restored from the volumes placed in the device specified on the DEV parameter.

volume-identifier: Specify the volume identifiers of the tapes or diskettes to be used for restoring the file. For each tape or diskette volume name, up to 6 characters can be specified using any combination of letters and digits. Up to 50 volume identifiers can be specified.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

Note: This parameter is ignored if a diskette device is specified in the DEV parameter.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

PHYFILE

Specifies the qualified name of the database physical file that is used as the input file for the restore operation.

The file must have a record length of between 40 and 120 bytes if it contains members saved in record mode. The record length must be 256 if it contains members saved on a System/36 in sector mode.

If the specified file does not exist or is not a physical file, no library members are restored. If the physical file contains several members, the first member of the file is used.

The name of the physical file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the database physical file that is used as the input file for the restore operation.

Examples

Example 1: Restoring All Members

```
RSTS36LIBM  TOMBR(XYZ1)  TOLIB(JOHNSON)  DEV(I1)
  SRCMBRS(*PRC)  MBROPT(*ALL)  FROMLABEL('XYZ1')
```

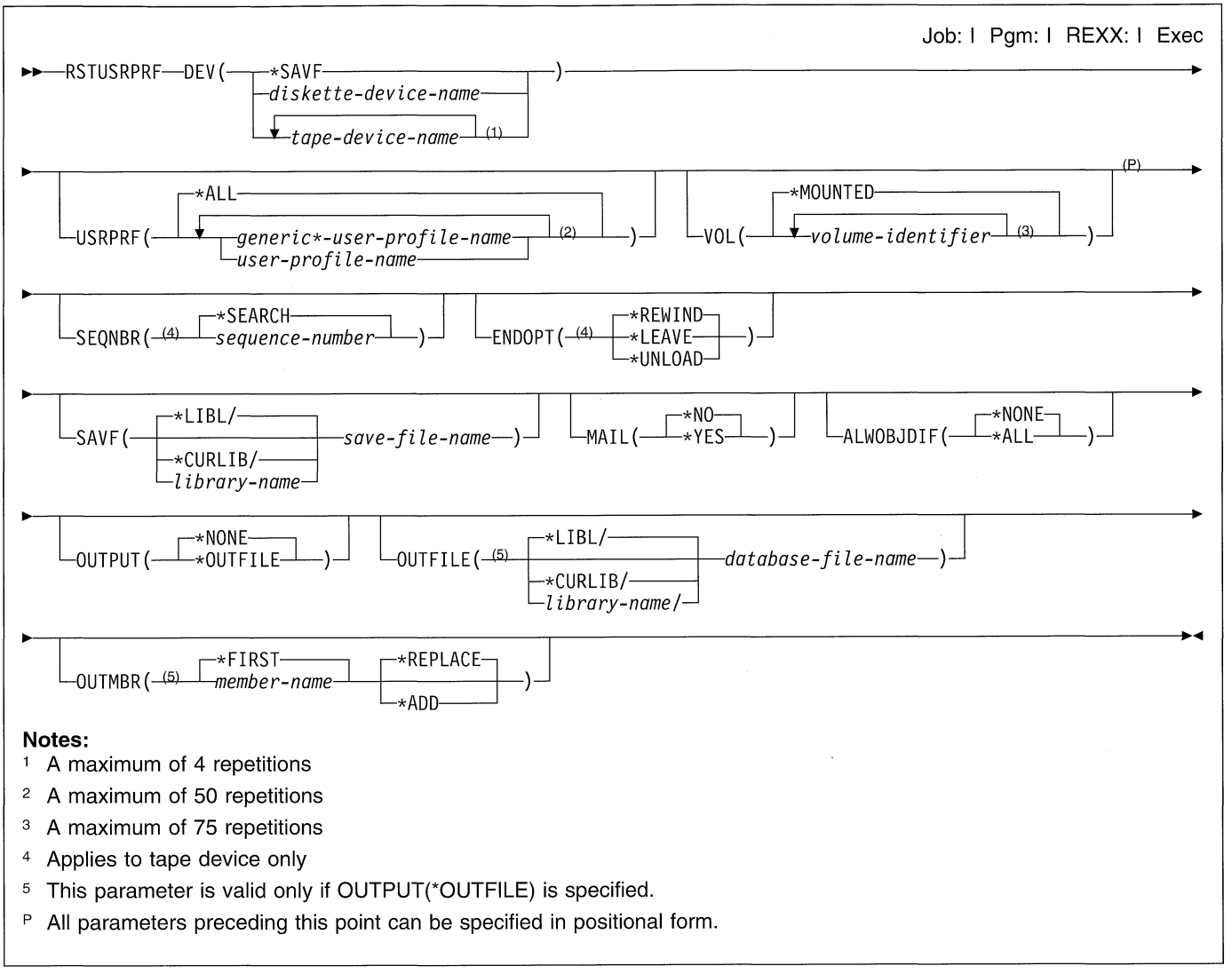
The single OCL procedure member XYZ1 is restored as a member of source file QS36PRC in library JOHNSON. Assuming I1 refers to a diskette device, the input diskette file must have the label XYZ1.

Example 2: Specifying a Generic Member Name

```
RSTS36LIBM  TOMBR(X*)  TOLIB(ORDER)  DEV(*PHYFILE)
  PHYFILE(NETLIB/S36SRC)
```

All source and procedure members with names starting with the character 'X' and that do not already exist as members of QS36SRC and QS36PRC in library ORDER are restored. The members are restored from file S36SRC in library NETLIB.

RSTUSRPRF (Restore User Profiles) Command



Notes:

1 A maximum of 4 repetitions

2 A maximum of 50 repetitions

3 A maximum of 75 repetitions

4 Applies to tape device only

5 This parameter is valid only if OUTPUT(*OUTFILE) is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Restore User Profiles (RSTUSRPRF) command restores the basic parts of a user profile or a set of user profiles saved by the Save System (SAVSYS) or the Save Security Data (SAVSECDTA) commands. The RSTUSRPRF command restores only the special authority given in the Create User Profile (CRTUSRPRF) command; it does not restore the authority for the named objects owned by other users. To do that, the Restore Authority (RSTAUT) command must be used after the profiles, libraries, and objects are restored. If all user profiles are restored, the authorization lists and authority holders that existed when the SAVSYS command or SAVSECDTA command was run are also restored.

Before this command is specified, all other operations on the system must be stopped. This requires ending all subsystems through the End Subsystem (ENDSBS(*ALL)) command or End System (ENDSYS) command or specifying this command when the operating system is started. The RSTUSRPRF command is normally used after the restore of the operating system but before the user libraries are restored. The user profiles must be restored before any libraries or objects belonging to them can be restored. After the libraries and their objects are restored, the authority for the objects is restored to the user profiles by the RSTAUT command. At the completion of the command, either message CPF3775 or message CPC3705 is sent to QHST. More information on restoring the system is in the *Basic Backup and Recovery Guide.*

The following situations may apply to user profiles being restored by the RSTUSRPRF command:

- If a user profile exists on the system, but not on the media, the system profile remains.
- If a user profile exists on the media, but not on the system, a new user profile is created.

- If the user profile exists on both the media and the system, the media user profile is restored.
- If the user profile exists on the media and is being restored individually, the new user profile is created without its password or group connection.
- If the user profile exists on both the media and the system, and it is being restored individually, the media user profile is restored. However, the password and group connection on the system remains unchanged.
- If all user profiles are being restored, the passwords and group connections are also restored from the media.
- If user profiles exist on the system, there are no changes to the existing object authorities.

Note: This command ignores all file overrides that are currently in effect for the job.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. The user of this command must have *SAVSYS authority in the user profile, specified by SPCAUT(*SAVSYS).
3. Before this command is specified, all other operations on the system must be ended. The End System (ENDSYS) or End Subsystem (ENDSBS) command can be used to end these operations. You must have *JOBCTL authority to use the ENDSYS or ENDSBS command.
4. To restore authorization lists and authority holders, USRPRF(*ALL) must be specified.

Required Parameter

DEV

Specifies the name of the diskette device, tape device, or save file used to restore the user profiles. The device names must already be known on the system by a device description. Up to four tape devices may be specified.

***SAVF:** The save file specified on the SAVF parameter is used for the restore operation.

diskette-device-name: Specify the name of the diskette device used to restore the user profiles.

tape-device-name: Specify the names of one or more tape devices used for the restore operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. Using more than one tape device permits one tape volume to be rewound and unloaded while another tape device processes the next tape volume.

Optional Parameters

USRPRF

Specifies the names of one or more user profiles to restore. To be restored, the user profiles must exist on the media from the Save System (SAVSYS) or Save Security Data (SAVSECDTA) commands.

***ALL:** All of the user profiles, authorization lists, and authority holders saved by the SAVSYS or SAVSECDTA commands are restored.

generic-user-profile-name:* Specify one or more generic names of sets of user profiles to restore. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk (*) substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix, for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

user-profile-name: Specify one or more names of specific user profiles that are restored. Both generic names and specific names can be specified in the same command.

VOL

Specifies the volume identifiers of the tape volumes or diskette volumes from which the objects are being restored. The volumes must be in the same order as they were when the library was saved. The volume that contains the beginning of the file to be restored should be placed in the tape or diskette unit. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: For tape, the user profiles are restored from the volumes placed on the tape devices, in the order that those devices were specified. This operation is normally performed immediately after a restore of the system is performed and the user profiles are on the same volume.

***MOUNTED:** The objects are restored from the volumes placed in the device specified on the DEV parameter.

volume-identifier: Specify the identifiers of up to 75 volumes in the order in which they are placed in the devices and used to restore the user profiles.

SEQNBR

Specifies the sequence number of the tape file used for the restore process.

***SEARCH:** The volume placed in the device is searched for a file containing the saved user profiles; when a match is found, the user profiles are restored. If a match is not found, you must load another tape and try the command again. If the last operation on the device specified ENDOPT(*LEAVE) (the tape is positioned at the location where the last operation ended), the file search starts with the first file beyond the current tape position. If ENDOPT(*LEAVE) was not used for the last operation (or if the tape was manually rewound since an ENDOPT(*LEAVE) operation), the search begins with the first file on the volume.

sequence-number: Specify the sequence number of the file.

ENDOPT

Specifies, when tape is used, what positioning operation is automatically done on the tape volume after the restore operation ends. If more than one tape volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

SAVF

Specifies the save file that contains the data from which to restore objects.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the file used to restore user profiles.

MAIL

Specifies whether the OfficeVision/400 distribution objects saved from a release before V2R2M0 are restored.

Note: You can specify *YES on this parameter only if you specify *ALL on the USRPRF parameter.

***NO:** Distribution objects that are part of your mail are not restored with the user profile.

***YES:** Distribution objects that are part of your mail are restored with the user profile if the save data was created before release V2R2M0. Otherwise, no distribution objects are restored. For saved distribution objects created on V2R2M0 or later, specify DLO(*MAIL) on the Restore Document Library Objects (RSTDLO) command to restore your mail.

ALWOBJDIF

Specifies whether certain differences encountered during a restore are allowed. There are two differences for this command:

- The owner of the object on the system is different than the owner of the object from the save.
- The object is secured by an authorization list and is being restored to a system other than the one on which it was saved.

Note: In order to use this parameter, *ALLOBJ authority is required.

***NONE:** The differences described above are not allowed for the restore operation. For an ownership difference, the object is not restored. For an authorization list difference, the object is restored, but the object is not linked to the authorization list, and public authority is set to *EXCLUDE.

***ALL:** The differences previously described are allowed for the restore operation. The object is restored. The following should be noted:

- If the owners of the object do not match, the object will be restored, but the ownership and authorities it had before the restore operation are retained.
- The informational message causes a diagnostic message to be sent indicating that security or integrity changes occurred during the restore. This results in an escape message for the restore function.
- If the user is restoring objects to a system different from the one on which they were saved and the objects are secured by an authorization list, specifying *ALL automatically links the objects to the authorization list again. If the authorization list does not exist on the new system, a message that includes the name of the missing list is issued.

OUTPUT

Specifies whether a listing that shows information about the status of the object is created and directed to an output file. The listing shows the restore information and shows all objects, restored, not restored, and excluded. Information about each object's security is listed for the restored objects. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***NONE:** No output is created.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: You must specify the database file name on the OUTFILE parameter when OUTPUT(*OUTFILE) is specified.

OUTFILE

Specifies the qualified name of the database file to which the information about the object is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QASRRSTO in QSYS with the format name QSRRST as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which the output of the command is directed.

OUTMBR

Specifies the name of the database file member to which the output is directed. If a member already exists, the system uses the second element of this parameter to determine whether the member is cleared before the new records are added. If the member does not exist and a member name is not specified, the system creates a member with the name of the output file specified on the OUTFILE parameter. If an output file member name is specified, but the member does not exist, the system creates it.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that receives the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it. If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

Examples

Example 1: Restoring All Profiles

```
RSTUSRPRF  DEV(TAP01)  SEQNBR(*SEARCH)  ENDOPT(*REWIND)
```

The saved version of all user profiles contained on the tape currently put on the tape drive named TAP01 is restored to the system. The tape is searched for the file, and the tape is rewound on completion or at the end of restore.

Example 2: Restoring Specific User Profiles

```
RSTUSRPRF  DEV(TAP01)  USRPRF(USRA USRB USRC USER*)
```

The saved version of all the user profiles must exist on the tape placed on tape drive TAP01. User profiles USRA, USRB, and USRC, along with all the user profiles whose names start with USER, are restored to the system.

Example 3: Restoring User Profiles from a Save File

```
RSTUSRPRF  DEV(*SAVF) USRPRF(USRX USRY)
  SAVF(QGPL/SAVESEC)
```

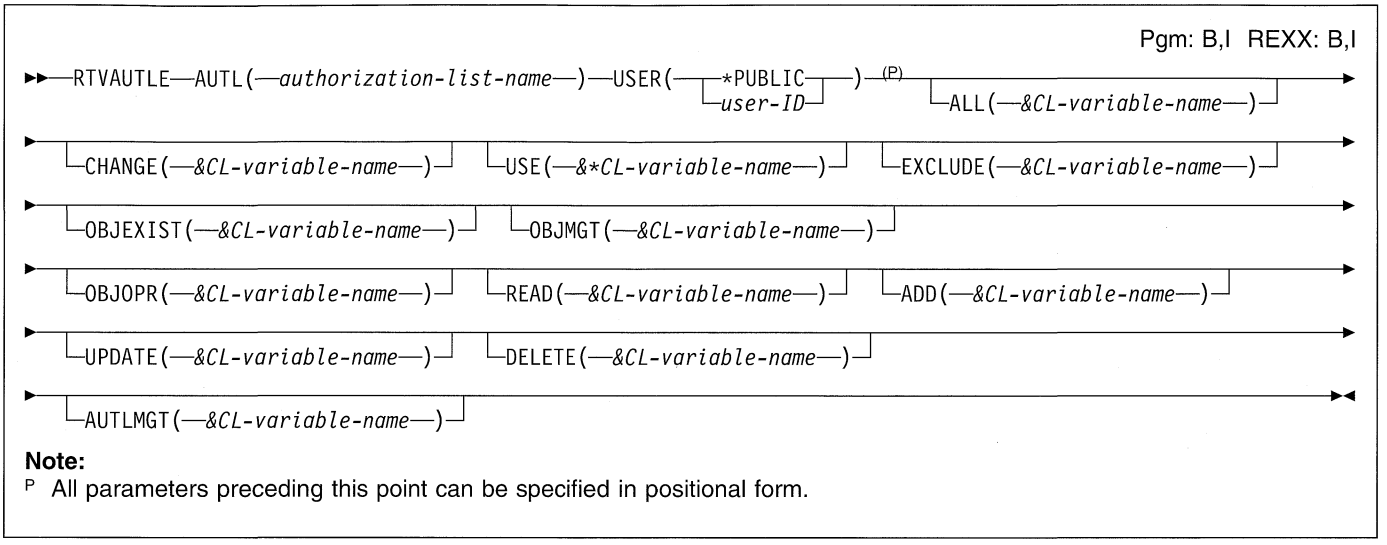
This command restores user profiles USRX and USRY to the system from the save file SAVESEC in library QGPL.

Example 4: Reporting Information about User Profiles Restored and Not Restored

```
RSTUSRPRF  DEV(TAP01) USRPRF(*ALL) OUTPUT(*OUTFILE)
  OUTFILE(PRFS92) OUTMBR(FOURQT *ADD)
```

This command restores all user profiles from the tape device TAP01. A list reporting information about user profiles restored and not restored is directed to the output file PRFS92. The output is received in the member FOURQT as an addition to existing information in the member.

RTVAUTLE (Retrieve Authorization List Entry) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Authorization List Entry (RTVAUTLE) command retrieves the authorities that a user has on the authorization list. It can be used with the Change Authorization List Entry command to change the user's authorities to include new authorities in addition to the existing authorities for the user.

The authorization list name and user name must be specified. The variables for each of the authorities the user might have are returned blank if the user does not have the authority; they are returned with the correct value for the Change Authorization List Entry (CHGAUTLE) command if the user has the authority. The values are returned in the specified variables for the specified user.

Users who have *AUTLMGT authority, who own the authorization list, or have *ALLOBJ authority can retrieve authority for any user on the list. Other users can retrieve their own authorities or the authority of *PUBLIC. The user must have *READ authority to the user profile from which the authorities are being read.

The following values are returned if the user has the authority specified by the keyword:

Keyword	Value
ADD	*ADD
ALL	*ALL
AUTLMGT	*AUTLMGT
CHANGE	*CHANGE
DELETE	*DLT
EXCLUDE	*EXCLUDE
OBJEXIST	*OBJEXIST
OBJMGT	*OBJMGT
OBJOPR	*OBJOPR
READ	*READ
UPDATE	*UPD
USE	*USE

The CL prompt for this command lists the minimum length for the variables next to the appropriate parameters to be retrieved. For character variables, a single number is shown.

Required Parameters

AUTL

Specifies the name of the authorization list that specifies the user's authorities.

USER

Specifies the name of the user whose information is being retrieved. If a variable is specified, it must be 10 characters in length and contain a user name or the value *PUBLIC.

***PUBLIC:** The information returned in the specified parameter is for the users who do not have any specific authority to the authorization list, and whose user group does not have any specific authority to the authorization list.

user-ID: Specify the user ID of the user whose information is being retrieved.

Optional Parameters

ALL

Specifies the name of a variable that is used to return *ALL if the user has *ALL authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *ALL authority.

CHANGE
Specifies the name of a variable that is used to return *CHANGE if the user has *CHANGE authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *CHANGE authority.

USE
Specifies the name of a variable that is used to return *USE if the user has *USE authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *USE authority.

EXCLUDE
Specifies the name of a variable that is used to return *EXCLUDE if the user has *EXCLUDE authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *EXCLUDE authority.

OBJEXIST
Specifies the name of a variable that is used to return *OBJEXIST if the user has *OBJEXIST authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *OBJEXIST authority.

OBJMGT
Specifies the name of a variable that is used to return *OBJMGT if the user has *OBJMGT authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *OBJMGT authority.

OBJOPR
Specifies the name of a variable that is used to return *OBJOPR if the user has *OBJOPR authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *OBJOPR authority.

READ
Specifies the name of a variable that is used to return *READ if the user has *READ authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *READ authority.

ADD
Specifies the name of a variable that is used to return *ADD if the user has *ADD authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *ADD authority.

UPDATE
Specifies the name of a variable that is used to return *UPD if the user has *UPD authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *UPD authority.

DELETE
Specifies the name of a variable that is used to return *DLT if the user has *DLT authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *DLT authority.

AUTLMGT
Specifies the name of a variable that is used to return *AUTLMGT if the user has *AUTLMGT authority. In CL programs, the variable has a length of 10 characters. Blanks are returned in the variable if the user does not have *AUTLMGT authority.

Example

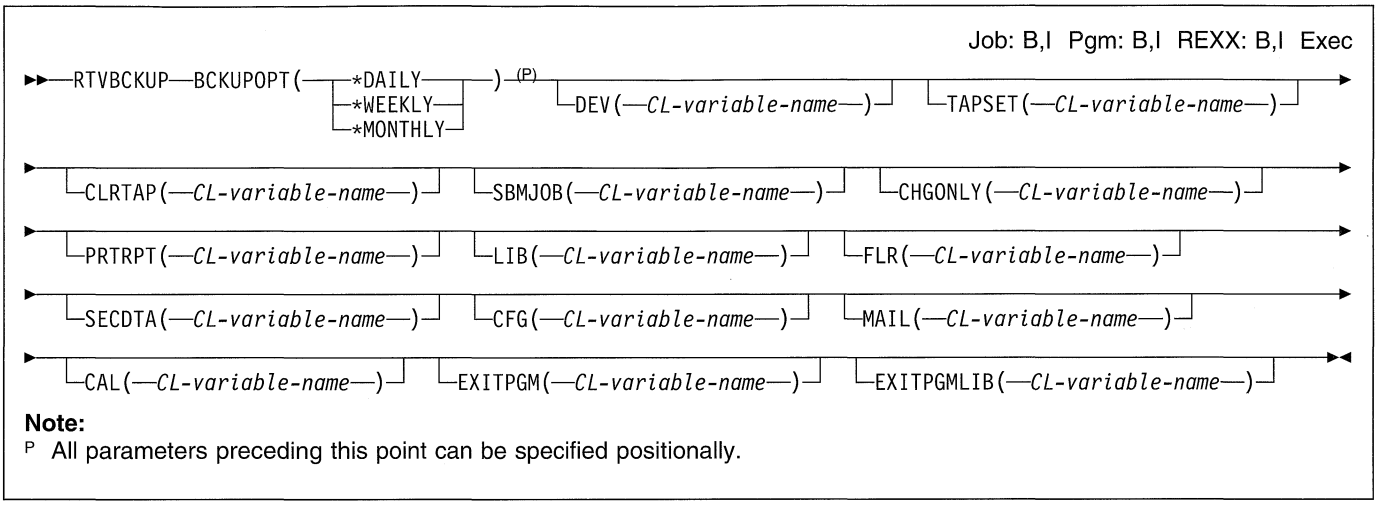
```
ADDAUTLE  AUTL(PAYROLL)  USER(TOM)
  AUT(*OBJOPR *READ *UPD *AUTLMGT)
```

When user Smith calls a CL program containing the following:

```
DCL     &CHG        *CHAR 10
DCL     &ALL        *CHAR 10
DCL     &USE        *CHAR 10
DCL     &EXCL       *CHAR 10
DCL     &OBJOP      *CHAR 10
DCL     &READ       *CHAR 10
DCL     &ADD        *CHAR 10
DCL     &UPD        *CHAR 10
DCL     &DLT        *CHAR 10
DCL     &AUTLM      *CHAR 10

RTVAUTLE  AUTL(PAYROLL)  USER(TOM)  USE(&USE)
  OBJOPR(&OBJOP)  AUTLMGT(&AUTLM)
```

This command retrieves the following authorities from the authorization list PAYROLL for user TOM: *USE, *OBJOPR, and *AUTLMGT. If TOM does not have the authority, blanks are returned.

RTVBCKUP (Retrieve Backup) Command



Note:
P All parameters preceding this point can be specified positionally.

Purpose

The Retrieve Backup (RTVBCKUP) command allows the user to retrieve the options in one of the predefined backups into CL variables. More information on backup is in the *Advanced Backup and Recovery Guide*.

Required Parameters

BCKUPOPT
Specifies the backup options to be retrieved.

***DAILY:** The daily backup options are retrieved.

***WEEKLY:** The weekly backup options are retrieved.

***MONTHLY:** The monthly backup options are retrieved.

Optional Parameters

DEV
Specifies the name of the CL variable that receives the device value. The variable has a minimum length of 43 characters. The value returned is a character string of four 10-character device names, separated by blanks.

TAPSET
Specifies the name of the CL variable that receives the tape set names. The variable has a minimum length of 34 characters (seven 4-character tape set names, separated by blanks).

CLRTAP
Specifies the name of the CL variable that receives the indicator for clearing the tape for backup. The variable must have a minimum length of 4 characters. The value returned is either *YES or *NO.

SBMJOB
Specifies the name of the CL variable that receives the indicator of whether the backup is run as a batch job. The variable must have a minimum length of 4 characters. The value returned is either *YES or *NO.

CHGONLY
Specifies the name of the CL variable that receives the indicator for saving changed objects only. The variable must have a minimum length of 4 characters. The value returned is either *YES or *NO.

PRTRPT
Specifies the name of the CL variable that receives the indicator for printing a report of saved objects. The variable must have a minimum length of 4 characters. The value returned is either *YES or *NO.

LIB
Specifies the name of the CL variable that receives the value specifying the libraries to save with this backup. The variable must have a minimum length of 10 characters. A value of *ALLUSR, *FROMLIST, or *NONE is returned.

FLR
Specifies the name of the CL variable that receives the value specifying the folders to save with this backup. The variable must have a minimum length of 10 characters. A value of *ALL, *FROMLIST, or *NONE is returned.

SECDTA
Specifies the name of the CL variable that receives the indicator for saving security data. The variable must have a minimum length of 4 characters. The value returned is either *YES or *NO.

CFG
Specifies the name of the CL variable that receives the indicator for saving configuration data. The variable must have a minimum length of 4 characters. The value returned is either *YES or *NO.

MAIL
Specifies the name of the CL variable that receives the indicator for saving OfficeVision/400 mail. The variable must have a minimum length of 4 characters. The value returned is either *YES or *NO.

CAL
Specifies the name of the CL variable that receives the indicator for saving OfficeVision/400 calendars. The variable must have a minimum length of 4 characters. The value returned is either *YES or *NO.

EXITPGM
Specifies the name of the CL variable that receives the name of the user program to call before and after the backup is run. The variable must have a minimum length of 10 characters. If no exit program is specified, *NONE is returned.

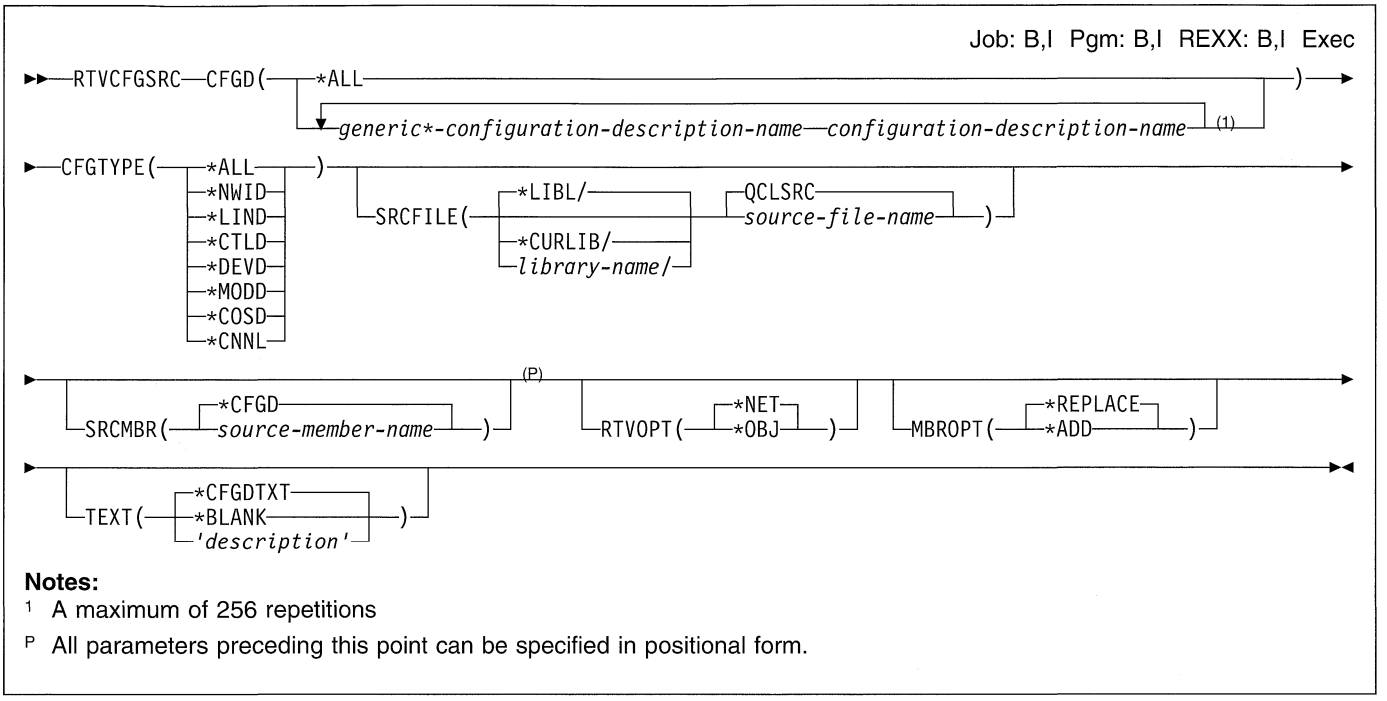
EXITPGMLIB
Specifies the name of the CL variable that receives the name of the library that contains the exit program. The variable must have a minimum length of 10 characters. If no exit program is specified, blanks are returned. If *LIBL is returned, the program uses the library list.

Example

```
RTVBCKUP  BCKUPOPT(*DAILY)  SBMJOB(&SBMJOBVAR)
  LIB(&LIBVAR)
```

This command retrieves the SBMJOB and LIB values for the daily backup into the CL variables SBMJOBVAR and LIBVAR respectively.

RTVCFGSRC (Retrieve Configuration Source) Command



Notes:

1 A maximum of 256 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Configuration Source (RTVCFGSRC) command retrieves CL command source to create existing configuration objects. This command allows easy portability of configuration objects from one system to another and provides a method of saving configurations without having to run the SAVSYS command.

Required Parameters

CFGD

Specifies the configuration objects of the CL source being retrieved. Multiple names can be specified; generic names are accepted. The CFGTYPE parameter specifies the type of object to which the names refer.

***ALL:** All objects specified by the CFGTYPE parameter are retrieved.

generic-configuration-description-name:* Specify the generic name of the configuration description name. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be retrieved only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

configuration-description-name: Specify the user-defined name of the configuration description.

CFGTYPE

Specifies the object type to which the names in the CFGD parameter are referring.

***ALL:** All network interfaces, connection lists, lines, controllers, devices, modes, and classes-of-service matching the specified names are retrieved in the following order:

1. Connection Lists
2. Network Interfaces
3. Non-TDLC line descriptions
4. Non-TDLC controller descriptions
5. TDLC line descriptions
6. TDLC controller descriptions
7. Device descriptions
8. Mode descriptions
9. Class-of-service descriptions
10. SWTCTLLST for line descriptions
11. SWTLINLST for controller descriptions
12. SWTNWILST for line descriptions
13. Printer for remote displays

***NWID:** All network interfaces matching the specified names are retrieved.

***LIND:** All lines matching the specified names are retrieved.

***CTLD:** All controllers matching the specified names are retrieved.

***DEVD:** All devices matching the specified names are retrieved.

***MODD:** All modes matching the specified names are retrieved.

***COSD:** All classes-of-service matching the specified names are retrieved.

***CNNL:** All connection lists matching the specified names are retrieved.

Optional Parameters

SRCFILE

Specifies the name of the previously created database source file where the retrieved source is written.

The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QCLSRC: The retrieved source is written in the QCLSRC source file.

source-file-name: Specify the name of the source file where the retrieved source is being written.

SRCMBR

Specifies the member name in the source file where the retrieved source is written. If the member does not exist, it is created.

***CFGD:** If *ALL, a generic name or list of names is specified on the CFGD parameter, the name of the source file member is CFGSRC. Otherwise, the name specified on the CFGD parameter is used as the member name.

source-member-name: Specify the source file member name.

RTVOPT

Specifies which attachment information is retrieved for the specified objects.

This parameter cannot be specified if CFGTYPE(*ALL) is specified. If CFGTYPE(*ALL) is specified, the source is retrieved for objects of all types with names as specified on the CFGD parameter, followed by switched attachment information (SWTCTLLST, SWTNWILST, and SWTLINLST parameters on Change Line, Change Network Interface, and Change Controller commands).

***NET:** The source for the configuration object names specified on the CFGD parameter, of the type specified on the CFGTYPE parameter are retrieved. Also retrieved are the downward-attached configuration objects. For example, if CFGTYPE(*CTLD) is specified, the configuration source for devices attached to the specified controllers is also retrieved. For CFGTYPE(*CTLD), the switched attachment configuration information (if any) is retrieved in the form of Change Controller (CHGCTLxxx) commands including the SWTLINLST parameter following the Create Controller (CRTCTLxxx) commands.

***OBJ:** The source for the configuration object names specified on the CFGD parameter, of the type specified on the CFGTYPE parameter, is retrieved.

MBROPT

Specifies whether the new records replace or are added to the existing records.

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

TEXT

Specifies text that briefly describes the configuration description. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***CFGDTXT:** If *ALL is specified on the CFGD parameter or a generic name or list of names, then *BLANK is used as the text. If MBROPT(*ADD) is specified, the member's text is replaced; otherwise, the text of the configuration object being retrieved is used as the text.

***BLANK:** Text is not specified.

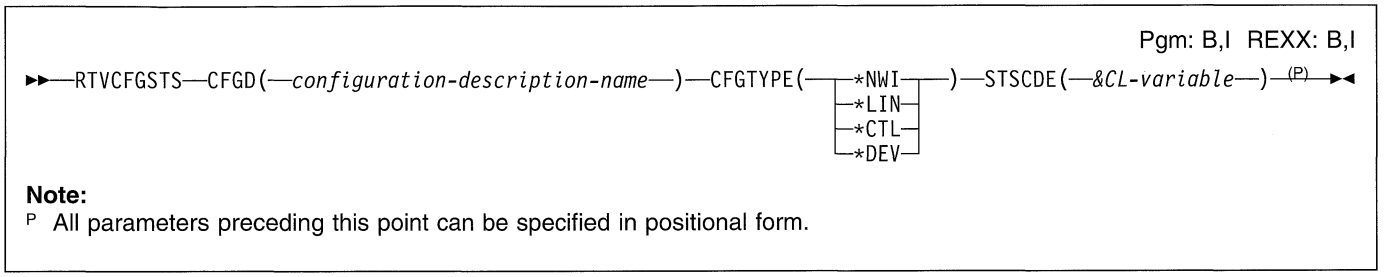
'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Example

```
RTVCFGSRC  CFGD(CTL*)  CFGTYPE(*CTLD)
  SRCMBR(CTLS)  RTVOPT(*OBJ)
```

This command places CL source statements in the file member CTLS in the source file QCLSRC. These source statements can be used to re-create object descriptions for all existing controllers with names beginning with CTL.

RTVCFGSTS (Retrieve Configuration Status) Command



Note:
(P) All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Configuration Status (RTVCFGSTS) command retrieves the status of a configuration object for use in a CL program.

Required Parameters

CFGD

Specifies the name of the configuration object whose status is retrieved. Valid values range from 1 through 10 characters.

CFGTYPE

Specifies the type of configuration object.

***NWI:** The object is a network interface description.

***LIN:** The object is a line description.

***CTL:** The object is a controller description.

***DEV:** The object is a device description.

STSCDE

Specifies the name of the variable that will contain the retrieved status. In a CL program, the variable name should be a decimal variable of length (5 0).

The possible values which can be returned include:

Value	Definition
0	**VARIED OFF**–The system is not using the description.
10	**VARY OFF PENDING**–The description is being varied off. During this time the system may be taking down the tasks which managed the resource.
20	**VARY ON PENDING**–The description is being varied on. During this time the system may be putting tasks in place to manage the resource, downloading microcode to an I/O processor, communicating with data circuit-terminating equipment (DCE), and so on.
30	**VARIED ON**–The tasks that manage the network interface, line, controller, or device have been put in place by the system, and the system has the capability to communicate with it.
40	**CONNECT PENDING**–This status is only valid for switched IDLC, SDLC, BSC, or asynchronous lines. The line is in this status while waiting for the switched connection to be established; this connection can be either a dial or an answer.
50	**SIGNON DISPLAY**–This status is only valid for display devices. The system is preparing the device to receive the sign-on display, sending the sign-on display, or the actual sign-on display is at the display station.
60	**ACTIVE**–The object is successfully placed in VARIED ON status. In addition: for network interfaces, one or more attached lines are in VARY ON PENDING status or higher; for lines, one or more attached controllers is in a VARY ON PENDING status or higher; for controllers, one or more attached devices is in a VARY ON PENDING status or higher; for devices, active status varies depending on the type of device. More information is in the *OS/400* Communications Configuration Reference* manual. A display device which is at a second sign-on display as a result of pressing the system request key will be considered ACTIVE.
70	**HELD**–This status is only valid for Device Descriptions. The user or the system held the communications device to prevent it from communicating. The Release Communications Device (RLSCMNDEV) command can be used to release the device.
80	**RCYPND**–Error recovery is pending for the network interface, line, controller, or device. A message indicating what error occurred appears on the QSYSOPR message queue.
90	**RCYCNL**–Error recovery is canceled for the network interface, line, controller, or device. An error occurred, and the operator replied with a C (to cancel error recovery) to a message, or the operator used a command (ENDNWIRCY, ENDLINRCY, ENDCTLRCY, ENDDEVRCY) to end error recovery.

100 **FAILED**–An error occurred for the network interface, line, controller, or device that can be recovered only by varying off and on again.

110 **DIAGNOSTIC MODE**–The network interface, line, controller, or device resource is being used by problem analysis procedures to diagnose problems, and the resource cannot be used by other users.

111 **DAMAGED**–The network interface, line, controller, or device description is damaged. This is a system error condition. Information indicating when this damage occurred appears in the history log (QHST). The description must be deleted and created once more before it can be used again.

112 **LOCKED**–The actual status of the resource cannot be determined because another job has an exclusive lock on the description. Retry at a later time, or use the Work with Object Lock (WRKOBJLCK) command to determine which job has the lock on the description.

113 **UNKNOWN**–The status indicator of the description cannot be determined. This is a system error condition. Use the Dump Object (DMPOBJ) command to dump the contents and/or attributes of the description to a spooled printer file, and contact an IBM representative.

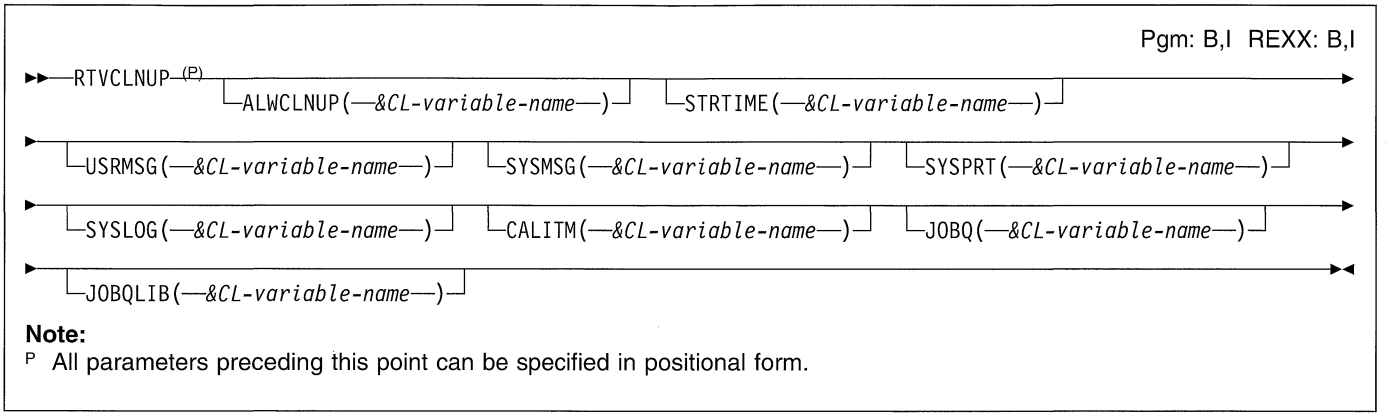
More information is in the *Communications Management Guide* and in the *OS/400* Communications Configuration Reference* manual.

Example

```
RTVCFGSTS  CFGD(ND01)  CFGTYPE(*LIN)
  STSCDE(&STSCODE)
```

This command retrieves the configuration status of the line configuration description ND01 for use in the CL variable &STSCODE.

RTVCLNUP (Retrieve Cleanup) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Cleanup (RTVCLNUP) command is used to retrieve a cleanup operation value for use in a CL or REXX program. The value is returned (copied) to the specified CL variable in the program.

Optional Parameters

ALWCLNUP

Specifies the name of the CL variable that receives the allow cleanup value. The variable named has a minimum length of 4 characters. *YES is returned if the cleanup operation is allowed to run. Otherwise, *NO is returned.

STRTIME

Specifies the name of the CL variable that receives the time the cleanup operation starts each day. The variable named has a minimum length of 10 characters. The special value *NONE or *SCDPWROFF, or the start time is returned.

USRMSG

Specifies the name of the CL variable that receives the value for deleting user messages on user profile message queues. The variable named has a minimum length of 5 characters. The special value *KEEP or the number of days user messages are kept before they are deleted is returned.

SYSMSG

Specifies the name of the CL variable that receives the value for deleting messages on the QSYSOPR message queue and on work station message queues. The variable named has a minimum length of 5 characters. The special value *KEEP or the number of days system messages are kept before they are deleted is returned.

SYSPRT

Specifies the name of the CL variable that receives the value for deleting job logs and other system output. The variable named has a minimum length of 5 characters. The special value *KEEP or the number of days job logs are kept before they are deleted is returned.

SYSLOG

Specifies the name of the CL variable that receives the value for deleting system journals, history files, problem log files, alert database, and program temporary fixes. The variable named has a minimum length of 5 characters. The special value *KEEP or the number of days system journals and system logs are kept before they are deleted is returned.

CALITM

Specifies the name of the CL variable that receives the value for deleting OfficeVision/400 calendar items. The variable named has a minimum length of 5 characters. The special value *KEEP or the number of days calendar items are kept before they are deleted is returned.

JOBQ

Specifies the name of the CL variable that receives the name of the job queue to which the cleanup batch jobs are submitted. The variable named has a minimum length of 10 characters. The name of the job queue under which cleanup batch jobs are run is returned.

JOBQLIB

Specifies the name of the CL variable that receives the library name of the job queue to which the cleanup batch jobs are submitted. The variable named has a minimum length of 10 characters. The name of the library where the job queue is located is returned.

Examples

Example 1: Retrieving Number of Days Messages are Kept

```
DCL  VAR(&UMSGDAYS)  TYPE(*CHAR)  LEN(5)
RTVCLNUP   USRMSG(&UMSGDAYS)
```

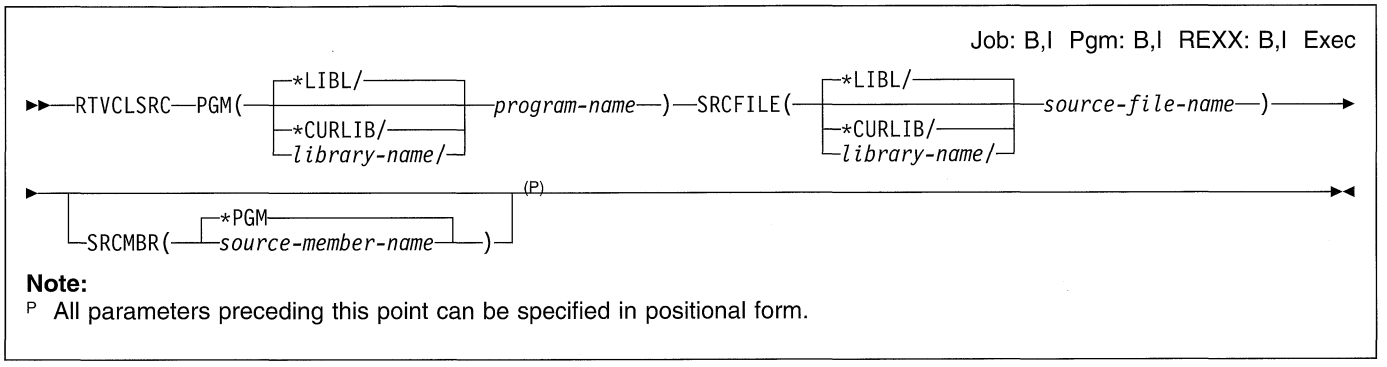
These commands retrieve the number of days that user messages are kept before being deleted.

Example 2: Retrieving Time Cleanup Operation Starts

```
DCL  VAR(&CLNUPTIME)  TYPE(CHAR)  LEN(10)
RTVCLNUP   STRTIME(&CLNUPTIME)
```

These commands retrieve the time that the cleanup operation starts.

RTVCLSRC (Retrieve CL Source) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve CL Source (RTVCLSRC) command is used to retrieve the source statements from a CL program used to compile that program. These source statements are placed into a source file member, which can be used as input when recompiling the CL program.

Required Parameters

PGM

Specifies the qualified name of the CL program whose source statements are being retrieved.

| The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of the CL program whose source statements are being retrieved.

SRCFILE

Specifies the qualified name of the previously created database source file into which the CL source statements are being written.

| The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

source-file-name: Specify the name of the database source file that contains the CL source statements.

Optional Parameters

SRCMBR

Specifies the name of the database source file member into which the CL source statements are being written. If not specified, the CL program name is assumed. If the member existed before running the command, it is cleared before any source statements are written into it. If the member did not exist, it is created.

***PGM:** The name of the CL program is used as the member name.

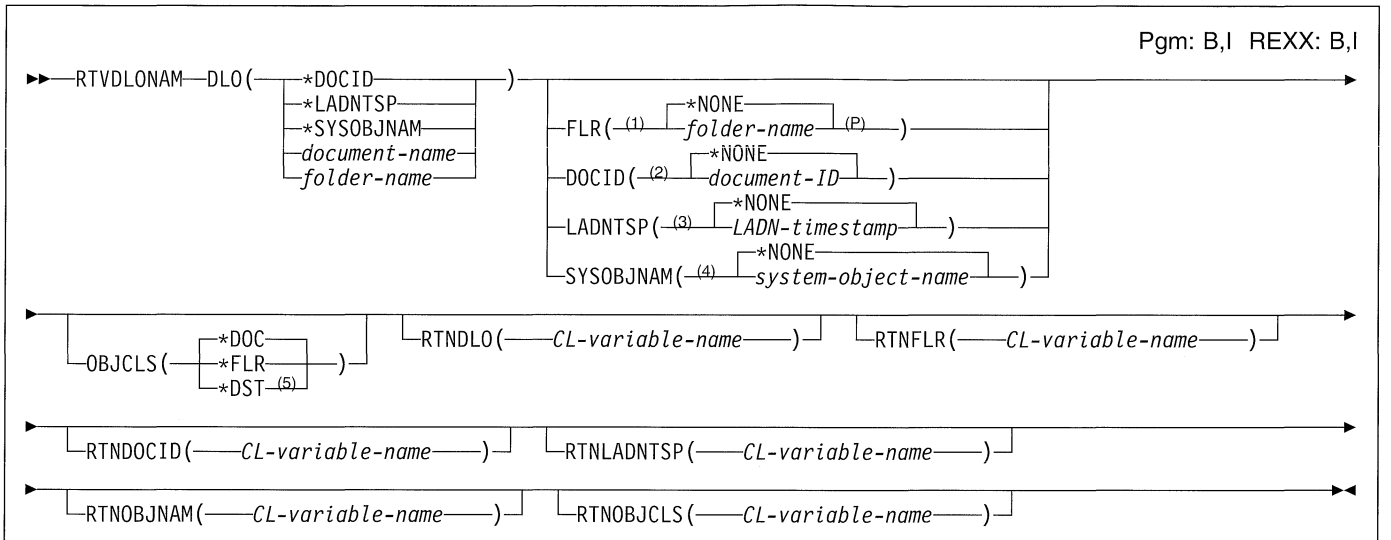
source-member-name: Specify the name of the source file member that contains the CL source statements.

Example

```
RTVCLSRC   PGM(JOHN1/TEXT1)  SRCFILE(JOHN2)
  SRCMBR(JOHN3)
```

This command retrieves the source statements from the CL program named TEXT1 in library JOHN1. The retrieved source statements are placed into the file named JOHN2, and are named as member JOHN3.

RTVDLONAM (Retrieve Document Library Object Name) Command



Notes:

1 FLR (*NONE) must be specified if *DOCID, *LADNTSP, or *SYSOBJNAM is specified on the DLO parameter.

P All parameters preceding this point can be specified in positional form.

2 Valid only if DLO(*DOCID) is specified.

3 Valid only if DLO(*LADNTSP) is specified.

4 Valid only if DLO(*SYSOBJNAM) is specified.

5 Valid only if DLO(*LADNTSP) or DLO(*SYSOBJNAM) is specified.

Purpose

The Retrieve Document Library Object Name (RTVDLONAM) command retrieves a list of all forms of a name for a folder, filed document, or distribution document.

Restrictions:

1. A user must have *USE authority to the filed document or folder to retrieve the various forms of the name.
2. Users must have *ALLOBJ authority to retrieve the various forms of the name for a distribution document.

Required Parameters

DLO

Specifies the name of the document or folder for which forms the name are retrieved.

***DOCID:** The library-assigned name specified on the DOCID parameter is used to identify the document or folder.

***LADNTSP:** The timestamp from the library-assigned document name (LADN) specified on the LADNTSP parameter is used to identify the document or folder.

***SYSOBJNAM:** The system object name specified on the SYSOBJNAM parameter is used to identify the document or folder.

document-name: Specify the user-assigned name of the document.

folder-name: Specify the user-assigned name of the folder.

Optional Parameters

FLR

Specifies the name of the folder where the object specified on the DLO parameter is located.

***NONE:** The name of folder that contains the object is not specified, the object is not contained in a folder, or the object is specified using the DOCID, LADNTSP, or SYSOBJNAM parameter.

folder-name: Specify the name of the folder that contains the object.

Note: FLR(*NONE) must be specified if the object is a first-level folder.

DOCID

Specifies the library-assigned name of the document or folder.

***NONE:** The object is not identified using its library-assigned name.

document-ID: Specify the library-assigned name of the document or folder object. Library-assigned names are

16 to 24 characters in length in the format YYYYMMDDHHMNSSHSSNSNSNSN, where:

YYYY = year
MM = month
DD = day
HH = hour
MN = minute
SS = second
HS = hundredths of a second
SNSNSNSN = system name (from 1 through 8 characters in length or omitted)

LADNTSP
Specifies the LADN timestamp of the document or folder.

***NONE:** The object is not identified using its LADN timestamp.

LADN-timestamp: Specify the LADN timestamp of the document or folder. The LADN timestamp is 16 hexadecimal characters in length in the format YYYYMMDDHHMNSSHS, where:

YYYY = year
MM = month
DD = day
HH = hour
MN = minute
SS = second
HS = hundredths of a second

SYSOBJNAM
Specifies the system object name.

***NONE:** The object is not identified using its system object name.

system-object-name: Specify the 10-character system object name of the document or folder

OBJCLS
Specifies the class of the object to locate.

***DOC:** The object is a filed document.

***FLR:** The object is a folder.

***DST:** The object is a distribution document.

RTNDLO
Specifies the name of a 12-character CL variable used to retrieve the user-assigned name of the selected document or folder. The *NONE value is returned for a distribution document or a document without a folder.

RTNFLR
Specifies the name of a 63-character CL variable used to retrieve the folder path of the selected document or folder. The *NONE value is returned for a distribution document, a document without a folder, or a first-level folder.

RTNDOCID
Specifies the name of a 24-character CL variable used to retrieve the library-assigned document name of the selected object. The variable is in the format YYYYMMDDHHMNSSHSSNSNSNSN. The *NONE value is returned for a distribution document.

RTNLADNTSP
Specifies the name of a 16-character CL variable used to retrieve the timestamp from the LADN of the selected object. The variable is in the format YYYYMMDDHHMNSSHS.

RTNOBJNAM
Specifies the name of a 10-character CL variable used to retrieve the system object name of the selected object.

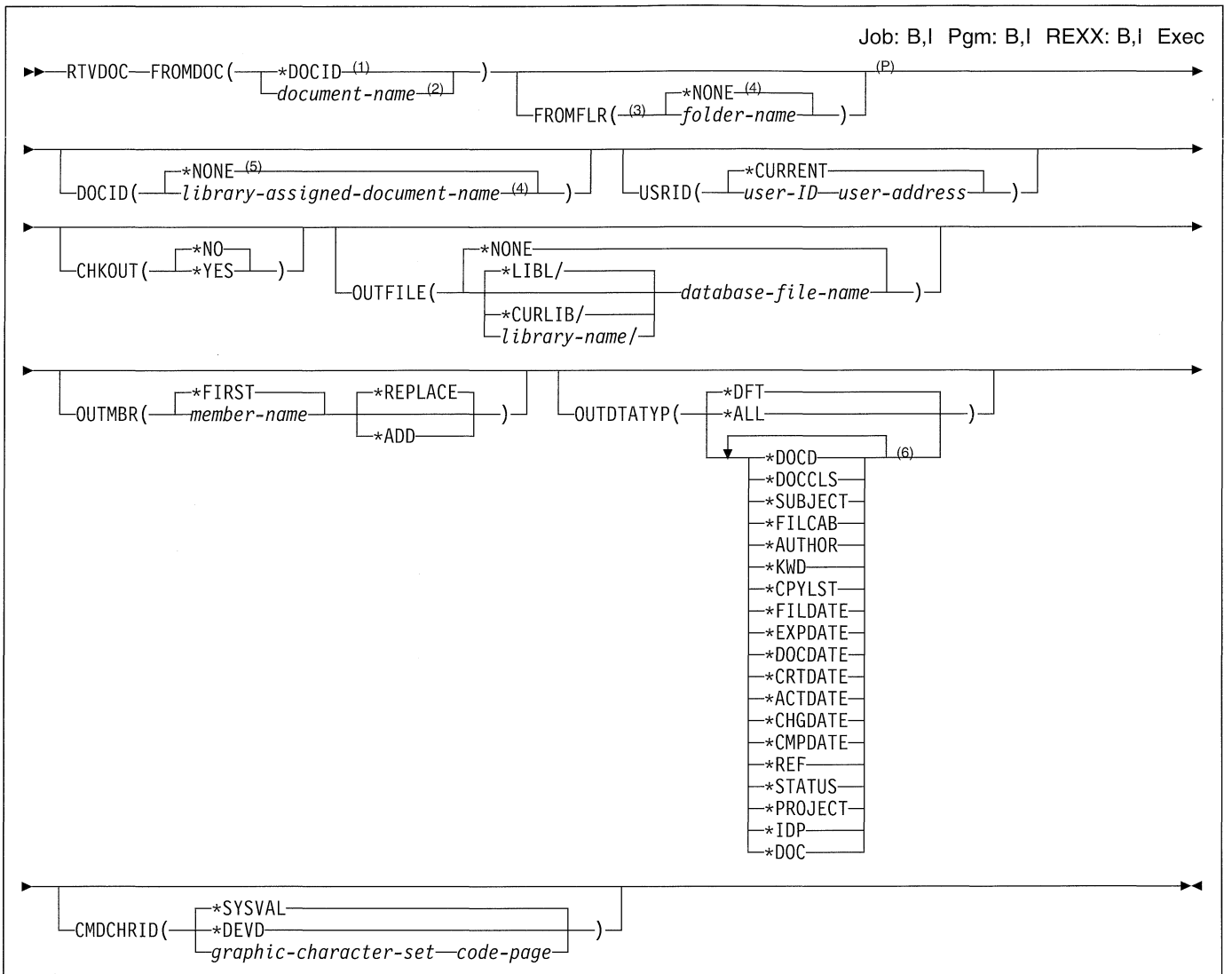
RTNOBJCLS
Specifies the name of a 8-character CL variable used to retrieve the object class. A value of *DOC is returned for a filed document, *FLR for a folder, and *DST for a distribution document.

Example

```
RTVDLONAM   DLO(MYDOC)  FLR(MYFLR)  OBJCLS(*DOC)
  RTNDOCID(&DOCID)
```

This command finds the document MYDOC in folder MYFLR and returns its document identifier in the variable &DOCID.

RTVDOC (Retrieve Document) Command



Notes:

1 FROMFLR(*NONE) and DOCID(*library-assigned-document-name*) must be specified if this value is specified.

2 FROMFLR(*folder-name*) and DOCID(*NONE) must be specified if this value is specified.

3 This parameter must be specified if a document name is specified on the FROMDOC parameter.

4 This value must be specified when FROMDOC(*DOCID) is specified.

5 This value must be specified when FROMDOC(*document-name*) is specified.

| P All parameters preceding this point can be specified in positional form.

6 A maximum of 19 repetitions

Purpose

The Retrieve Document (RTVDOC) command allows a user to copy information from a specific document to a database file and allows the user to check out a document from the document library. Documents whose storage has been freed (STG(*FREE) specified on the Save Document Library Object (SAVDLO) command) cannot be retrieved.

Restrictions:

1. To retrieve any records from the document to a database file, you must have *USE authority to the document or be working on behalf of a user that has *USE authority to the document.
2. To check out the document, you must have at least *CHANGE authority to the document, or be working on behalf of a user that has *CHANGE authority to the document.

3. To work on behalf of another user, you must have either *ALLOBJ authority or special permission (granted with the Grant User Permission (GRTUSRPMN) command).

Required Parameters

FROMDOC

Specifies the name of the document being retrieved.

***DOCID:** The document being retrieved is identified by the library-assigned document name specified on the DOCID parameter.

document-name: Specify the name of the document that is retrieved.

Optional Parameters

FROMFLR

Specifies the name of the folder that contains the document being retrieved.

***NONE:** No folder name is specified when the document is identified by the DOCID parameter.

folder-name: Specify the name of the folder that contains the document being retrieved.

DOCID

Specifies the library-assigned name of the document being retrieved. This is the name assigned to the document by the system when it is created. The library-assigned document names can be determined by using the Query Document Library (QRYDOCLIB) command. The library-assigned document name is also returned in a completion message when the File Document (FILDOC) command is used.

Library-assigned document names are 24 characters in length with the following format: YYYYMMDDHHMNSSHSSNSNSNSN where:

YYYY = year
MM = month
DD = day
HH = hour
MN = minute
SS = second
HS = hundredths of a second
SNSNSNSN = system name

***NONE:** No library-assigned document name is required when the document is identified by the FROMDOC parameter.

library-assigned-document-name: Specify the library-assigned document name sent. More information on library-assigned document names is in the *Distribution Services Network Guide*.

USRID

Specifies the user ID and address of the user for whom the request is made.

***CURRENT:** The user profile under which the current job is running is used.

Element 1: User ID

user-ID: Specify the user ID of the user for whom the document is retrieved.

Element 2: User Address

user-address: Specify the user address of the user for whom the document is retrieved.

CHKOUT

Specifies whether the document being retrieved can be replaced with new or changed data. If the document is read only, then specify CHKOUT(*NO). If the document being retrieved cannot be replaced, and CHKOUT(*YES) is specified, an error occurs.

Note: If a user is retrieving a document to check it out and a cancel request is entered, the document may or may not be checked out.

***NO:** The retrieve request only reads the data. Users requesting this function need only *USE authority to the document. The authority for the public is *READ authority.

***YES:** The document data can be updated and replaced later. Users requesting this function must have *CHANGE authority. The document is unavailable to other users until a replace operation is done using the Replace Document Command (RPLDOC) The checkout flag can be reset with the CHGDOCD or EDTDLOAUT command.

OUTFILE

Specifies the name of the database file where the document data of the display is directed. If the output file does not exist, this command creates a database file in the specified library. If the file is created by this function, the text "OUTFILE created by RTVDOC" is shown and the authority given to users with no specific authority is *EXCLUDE.

Note: The output file format must be the same as the OSRTVD of the system file QSYS/QAOSIRTV.

More information on defining the format of database files (output files) is in the *Office Services Concepts and Programmer's Guide*.

***NONE:** The output is not directed to a database file.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to receive the output.

This file can be reused when other RTVDOC commands are used. Output can be added to the file or can replace the existing records. The IBM-supplied database file QAOSIRTV in library QSYS cannot be specified.

OUTMBR

Specifies the name of the database file member to which the output is directed. If a member already exists, the system uses the second element of this parameter to determine whether the member is cleared before the new records are added. If the member does not exist and a member name is not specified, the system creates a member with the name of the output file specified on the OUTFILE parameter. If an output file member name is specified, but the member does not exist, the system creates it.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The system clears the existing member and adds the new records.

Note: Document data for more than one document can be added to the same member. However, an error occurs if the member is filed to another document with the FILDOC command. The FILDOC command only files the first document in the member and issues an error message.

***ADD:** The system adds the new records to the end of the existing records.

OUTDTATYP

Specifies what parts of information about the selected documents are written to the output file if one is specified.

***DFT:** The document information record is written to the output file. The following record codes are written to the output file:

Record Code	Description
105	Document Description
800	Document Data

***ALL:** All information records about the document are written to the output file.

Record Code	Description
010	Distribution Description
020	Message Text
105	Document Description
110	Creation Date
115	Expiration date
120	Document date
125	File date
130	Change date
135	Action due date
140	Completion date
145	Author
150	Copy list
155	Document class
160	File cabinet reference
165	Subject
170	Keyword
175	Reference
180	Status
185	Project
500	Interchange document profile data
800	Document Data

***DOCD:** The document information record is written to the output file.

***DOCCLS:** The document class record is written to the output file.

***SUBJECT:** The subject records are written to the output file.

***FILCAB:** The filing cabinet reference code is written to the output file.

***AUTHOR:** The author records are written to the output file.

***KWD:** The keyword records are written to the output file.

***CPYLST:** The copy list records are written to the output file.

***FILDATE:** The file date record is written to the output file.

***EXPDATE:** The expiration date record is written to the output file.

***DOCDATE:** The document date record is written to the output file.

***CRTDATE:** The create date record is written to the output file.

***ACTDATE:** The action due date record is written to the output file.

***CHGDATE:** The date last changed record is written to the output file.

***CMPDATE:** The completion date record is written to the output file.

***REF:** The reference record is written to the output file.

***STATUS:** The status record is written to the output file.

***PROJECT:** The project record is written to the output file.

***IDP:** The interchange document profile (IDP) is written to the output file.

***DOC:** The document data record is written to the output file.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the *Guide to Programming Displays*.

Note: This value translates the USRID parameter to character set and code page set of '930 256'. The *Communications: Distribution Services Network Guide* contains the character set and code page table for '930 256'.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values used to create the command parameter.

Element 2: Code Page

code-page: Specify the code page value used to create the command parameters. Valid values range from 1 through 999.

Examples

Example 1: Copying All Information

```
RTVDOC  FROMDOC(MYDOC)  FROMFLR(PERSONAL)
  USRID(*CURRENT)  OUTFILE(*CURLIB/MYFILE)
  OUTMBR(*FIRST)  MBROPT(*ADD)  OUTDTATYP(*ALL)
```

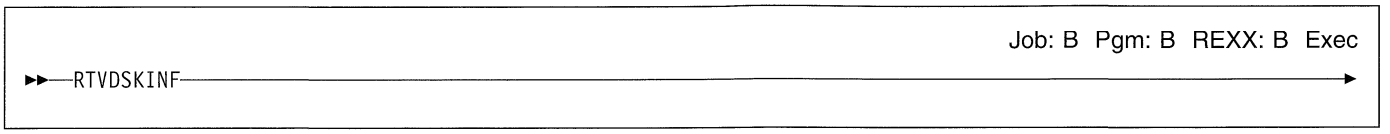
This command copies all information about document MYDOC located in folder PERSONAL for the current user of this command. CHECKOUT(*NO) is assumed; therefore, the document data can only be read. The output is directed to the database file MYFILE in the user's current library and is added to the first member in that file.

Example 2: Copying Default Information

```
RTVDOC  FROMDOC(SECOP)  FROMFLR(PERSONAL)
  USRID(MARY SYSTEM1 )  CHKOUT(*YES)
  OUTFILE(MARLIB/SECFILE)  OUTMBR(*FIRST *ADD)
```

This command copies the default information (*DOCD and *DOC) about document SECOP located in folder PERSONAL for MARY. The document can be updated with new data and then replaced. The current user of this command must have the authority to work on behalf of MARY given by Mary by using the GRTUSRPMN command. The output is directed to the database file SECFILE in Mary's library MARLIB. The output is added to the first member of SECFILE.

RTVDSKINF (Retrieve Disk Information) Command



Purpose

The Retrieve Disk Information (RTVDSKINF) command is used to collect disk space information. The collected information is stored in the library QUSRSYS, in a database file named QAEZDISK, in a member named QCURRENT.

Each time this command is run, existing information in QCURRENT is written over. To save existing information in QCURRENT, rename file QAEZDISK or copy it to another file.

Note: Do not rename member QCURRENT and leave it within file QAEZDISK. If there is more than one member in file QAEZDISK, the results of running this command can be unpredictable.

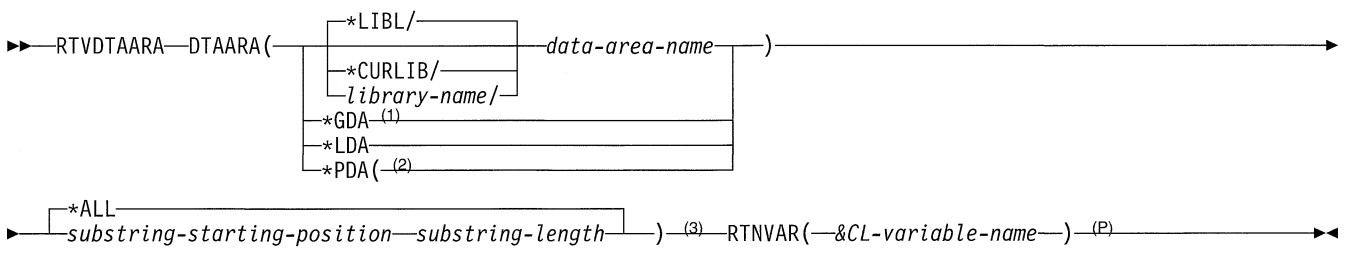
There are no parameters for this command.

Example

```
RTVDSKINF
```

This command retrieves disk space information and stores it in member QCURRENT of database file QAEZDISK. Any information in QCURRENT is overwritten.

RTVDTAARA (Retrieve Data Area) Command



Notes:

1 This value is valid only if this job is a group job.

2 This value is valid only if this job is a prestart job.

3 This option is allowed only if the data area TYPE is *CHAR.

P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Data Area (RTVDTAARA) command is used in a control language (CL) program to retrieve all or part of a specified data area and to copy it into a CL variable within the CL program. RTVDTAARA does not retrieve any other attributes of the data area. Existence of the data area is not required at the time the CL program is compiled.

If the job is a group job, the data area specified may be the group data area (*GDA). This data area is automatically associated with the group, and it is inaccessible from jobs outside the group. The length of this character data area is 512 bytes. More information about group jobs is in the *Work Management Guide*.

A local data area (*LDA) is a character data area that is 1024 bytes in length, and it is automatically associated with the job. Another job cannot access the local data area.

If the job is a prestart job, the data area specified may be the data area that contains program initialization parameter data (*PDA). This data area is automatically associated with the prestart job and is inaccessible from other jobs. The length of this character data area is 2000 bytes. More information about prestart jobs is in the *Work Management Guide*.

When a local data area or group data area must be retrieved during the processing of the RTVDTAARA command, the data area is locked during the retrieval operation so that commands in other jobs cannot change or destroy it until the operation is complete. If the data area is shared with other jobs and is updated in steps involving more than one command in a job, the data area should be explicitly allocated to that job until all the steps have been performed. A data area other than a local data area, group data area, or program initialization parameter data area can be explicitly allocated with the Allocate Object (ALCOBJ) command. No allocation is necessary for a local data area, group data area, or program initialization parameter data area.

Restrictions:

1. This command is valid only in compiled CL programs.
2. To use this command, the user must have *CHANGE authority for the data area and *USE authority for the library where the data area is located. No specific authority is required to retrieve the value of a local data area or group data area.

Required Parameters

DTAARA

Specifies the qualified name of the data area whose value is retrieved.

The name of the data area can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

Element 1: Data Area Name

data-area-name: Specify the name of the data area whose value is being retrieved.

Element 2: Data Area Options

***GDA:** The value of the group data area is being retrieved. This value is valid only if this is a group job.

***LDA:** The value of the local data area is being retrieved.

***PDA:** The value of the program initialization parameter data area is being retrieved. This value is valid only if this is a prestart job.

The substring retrieval option, which is valid for character data areas only, specifies the starting position and the length of the portion of the data area that is being retrieved into the CL character variable.

Element 3: Starting Position of the Data Area

***ALL:** The entire data area is retrieved.

substring-starting-position: Specify the starting position of the data area being retrieved.

Element 4: Length of the Data Area

substring-length: Specify the length of the data area substring being retrieved.

It is not possible to retrieve data outside the data area. The combination of starting position and length must always specify positions within the data area.

RTNVAR

Specifies the name of the CL program variable that receives the contents of the data area being returned.

No type conversion is performed by the RTVDTAARA command:

- If RTNVAR is declared as TYPE(*DEC), the data area retrieved must be TYPE(*DEC).
- If RTNVAR is declared as TYPE(*CHAR), the data area retrieved must be either TYPE(*CHAR) or TYPE(*LGL).
- If RTNVAR is declared as TYPE(*LGL), the data area retrieved must be either TYPE(*LGL) or TYPE(*CHAR) with a value of either '0' or '1'.

If a retrieved character string is shorter than the length of the variable specified by the RTNVAR parameter, the value is padded on the right with blanks. The retrieved string length must be less than or equal to the CL variable length.

When decimal data areas are retrieved, the decimals are aligned. The value of the integer portion of the data area must fit into the integer portions of the CL variable.

Fractional data is truncated if the fraction contains more digits than the CL variable.

Examples

Assume data area DA1 has been created by the following command:

```
CRTDTAARA   DTAARA(DA1)  TYPE(*CHAR)  LEN(3)  VALUE(ABC)
```

and variable &CLVAR1 has been declared as:

```
DCL   VAR(&CLVAR1)  TYPE(*CHAR) LEN(5)  VALUE(VWXYZ)
```

Example 1: Running This Command

```
RTVDTAARA   DTAARA(DA1)  RTNVAR(&CLVAR1)
```

results in:

```
&CLVAR1 = 'ABC  '
```

Example 2: Running This Command

```
RTVDTAARA   DTAARA(DA1 (2 1))  RTNVAR(&CLVAR1)
```

results in:

```
&CLVAR1 = 'B    '
```

Example 3: Decimal Data Area Example

Assume data area DA2 has been created with the following attributes:

```
CRTDTAARA   DTAARA(DA2)  TYPE(*DEC)  LEN(5 2)  VALUE(12.
```

and variable &CLVAR2 has been declared as:

```
DCL   VAR(&CLVAR2)  TYPE(*DEC)  LEN(5 1)  VALUE(4567.8)
```

Example 4: Running This Command

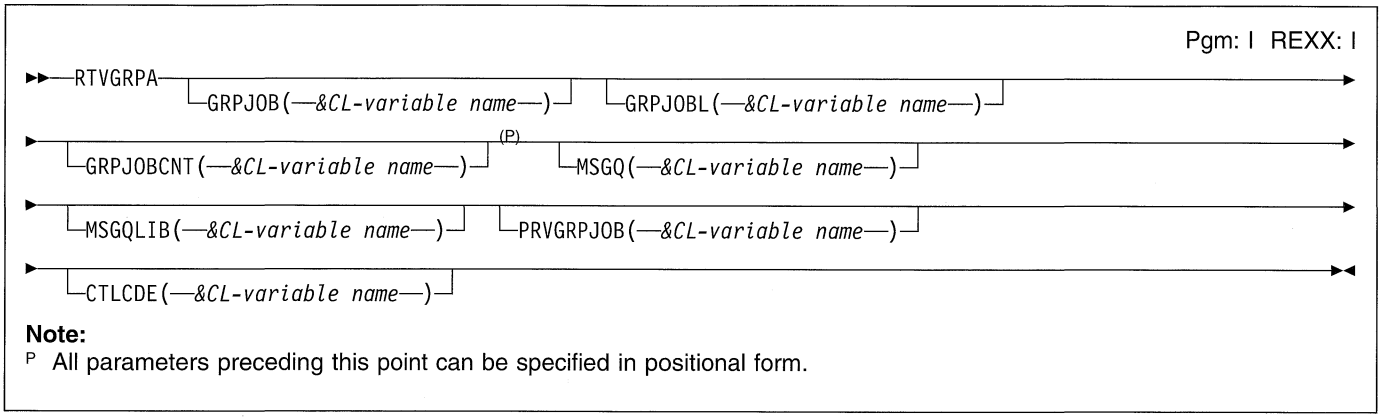
```
RTVDTAARA   DTAARA(MYLIB/&DTAARA)  RTNVAR(&CLVAR2)
```

results in:

```
CLVAR2 = 0012.3
```

Note that fractional digits are truncated instead of rounded.

RTVGRPA (Retrieve Group Attributes) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Group Attributes (RTVGRPA) command retrieves information about the group in which the job that issued the RTVGRPA command belongs. The following attributes can be retrieved:

- The group job name of the job calling the RTVGRPA command
- A list that contains information about all active jobs in the group
- A count of the number of active jobs in the group
- The name of the group message queue
- The library where the group message queue resides
- The group job name and job number of the previously active job in the group
- A control code indicating why the currently active job in the group gained control

The prompt for this command lists the minimum length for retrieved variables next to the appropriate parameters. For character variables, a single number is shown. For decimal variables, two numbers are shown. The first number indicates the minimum variable length, and the second number indicates the minimum number of decimal positions.

Optional Parameters

GRPJOB

Specifies the name of the CL variable that receives the group job name of the job. The variable must be a character variable with a minimum length of 10 characters. If the group job name has fewer characters than the variable allows, the value is padded on the right with blanks.

Note: This is the only parameter of the RTVGRPA command that is meaningful if the job that issued the RTVGRPA command is not a group job. When this occurs, the variable is set to the special value of *NONE, and other parameters specified in the command are filled with zeros or blanks (numeric variables with zeros, character variables with blanks). If this parameter is not specified and the job issuing this command is not a group job, an error message is sent.

GRPJOBL

Specifies the name of the CL variable that receives the list of jobs in the group. Each entry contains the job's group job name, the job number, and the 50 characters of descriptive text.

The maximum number of entries in the variable is 16. The entries are ordered by most recently active job (the group job that is issuing the RTVGRPA command is always first in the variable).

The variable must be a character variable with a minimum length of 1056 characters. If the group job list has fewer characters than the variable allows, the value is padded on the right with blanks.

The group job list has the following format:

Entry-1

Group-job-name	CHAR(10)
Job-number	CHAR(6)
Group-job-text	CHAR(50)

Entry-2

.
.
.

Entry-16

Group-job-name	CHAR(10)
Job-number	CHAR(6)
Group-job-text	CHAR(50)

This list contains all of the active jobs in the group. Jobs in the group that have not completely ended (jobs that have been canceled) are not listed.

GRPJOBCNT

Specifies the CL variable that receives the count of active jobs in the group. The CL variable must be a three-position decimal variable with no decimal positions.

The CL variable contains the number of nonblank entries in the group job list. The count includes all of the active jobs in the group. Jobs that have not completely ended (jobs that have been canceled) are not counted.

MSGQ

Specifies the name of the CL variable that receives the group message queue name. This must be a character variable with a minimum length of 10 characters. If the message queue name has fewer characters than the variable allows, the value is padded on the right with blanks.

MSGQLIB

Specifies the name of the CL variable that receives the name of the library that contains the group message queue. This variable must be a character variable with a minimum length of 10 characters. If the library name has fewer characters than the variable allows, the value is padded on the right with blanks. If there is no message queue associated with the group, the CL variable is set to blanks.

Refer to the Change Group Attributes (CHGGRPA) command description for more information on the use of the group message queue.

PRVGRPJOB

Specifies the name of the CL variable that receives the group job name and job number of the previously active job in the group. The variable must be a character variable with a minimum length of 16 characters. If the group job name has fewer characters than the variable allows, the value is padded on the right with blanks.

If there is no previously active job in the group (no Transfer to Group Job [TFRGRPJOB] command is issued), the group job name portion of the CL variable is set to the special value of *NONE, and the job number portion of the CL variable is set to blanks. The CL variable is returned in the following format:

```
Group-job-name      CHAR(10)
Job-number          CHAR(6)
```

Even though a group job name and job number are returned, it is not necessarily true that the group job still exists. There are cases in which the previously active job in the group has ended, which caused the currently active job to become active.

CTLCDE

Specifies the name of the CL variable that receives information about why the active job in the group has gained control. The CL variable must be a three-position decimal variable with no decimal positions. The following control codes (and their meanings) are possible:

0 There was no previously active job (no TFRGRPJOB commands have been run for this group).

10 The previously active job selected this job to be transferred to on the TFRGRPJOB command.

20 The previously active job's first group program ended normally, and this job was the most recently active job in the group.

30 The previously active job was ended by the End Group Job (ENDGRPJOB) command, and this job was selected to gain control (the RSMGRPJOB parameter specified this group job).

40 The previously active job was ended by the End Group Job (ENDGRPJOB) command and selected a job other than this job to gain control (which was ended before it could be resumed). Since this job was the most recently active job in the group, control is passed to it.

50 The previously active job was ended by the End Group Job (ENDGRPJOB) command, and this job was the most recently active job in the group (the RSMGRPJOB parameter specified *PRV).

60 The previously active job's first group program ended abnormally, and this job was the most recently active job in the group.

70 The previously active job was ended by the End Job (ENDJOB) command, and this job was the most recently active job in the group.

Examples

Assume jobs 0001/QUSER/WORKST01 and 0002/QUSER/WORKST01 are group jobs with group job names GROUPJ1 and GROUPJ2, respectively. Also assume that message queue QGPL/GROUPMSGQ is associated with the group. If group job GROUPJ1 has just issued the TFRGRPJOB command to transfer to group job GROUPJ2, and GROUPJ2 called the following CL program:

PGM Example

```
DCL VAR(&GRPJOBN)      TYPE(*CHAR)     LEN(10)
DCL VAR(&GRPJOBL)      TYPE(*CHAR)     LEN(1056)
DCL VAR(&GRPCOUNT)     TYPE(*DEC)      LEN(3 0)
DCL VAR(&MSGQNAME)     TYPE(*CHAR)     LEN(10)
DCL VAR(&MSGQLIB)      TYPE(*CHAR)     LEN(10)
DCL VAR(&PRVJOB)       TYPE(*CHAR)     LEN(16)
DCL VAR(&CTLCODE)      TYPE(*DEC)      LEN(3 0)

  RTVGRPA  GRPJOB(&GRPJOBN)  GRPJOBL(&GRPJOBL)
    GRPJOBCNT(&GRPCOUNT)  MSGQ(&MSGQNAME)
    MSGQLIB(&MSGQLIB)  PRVGRPJOB(&PRVJOB)
    CTLCDE(&CTLCODE)
```

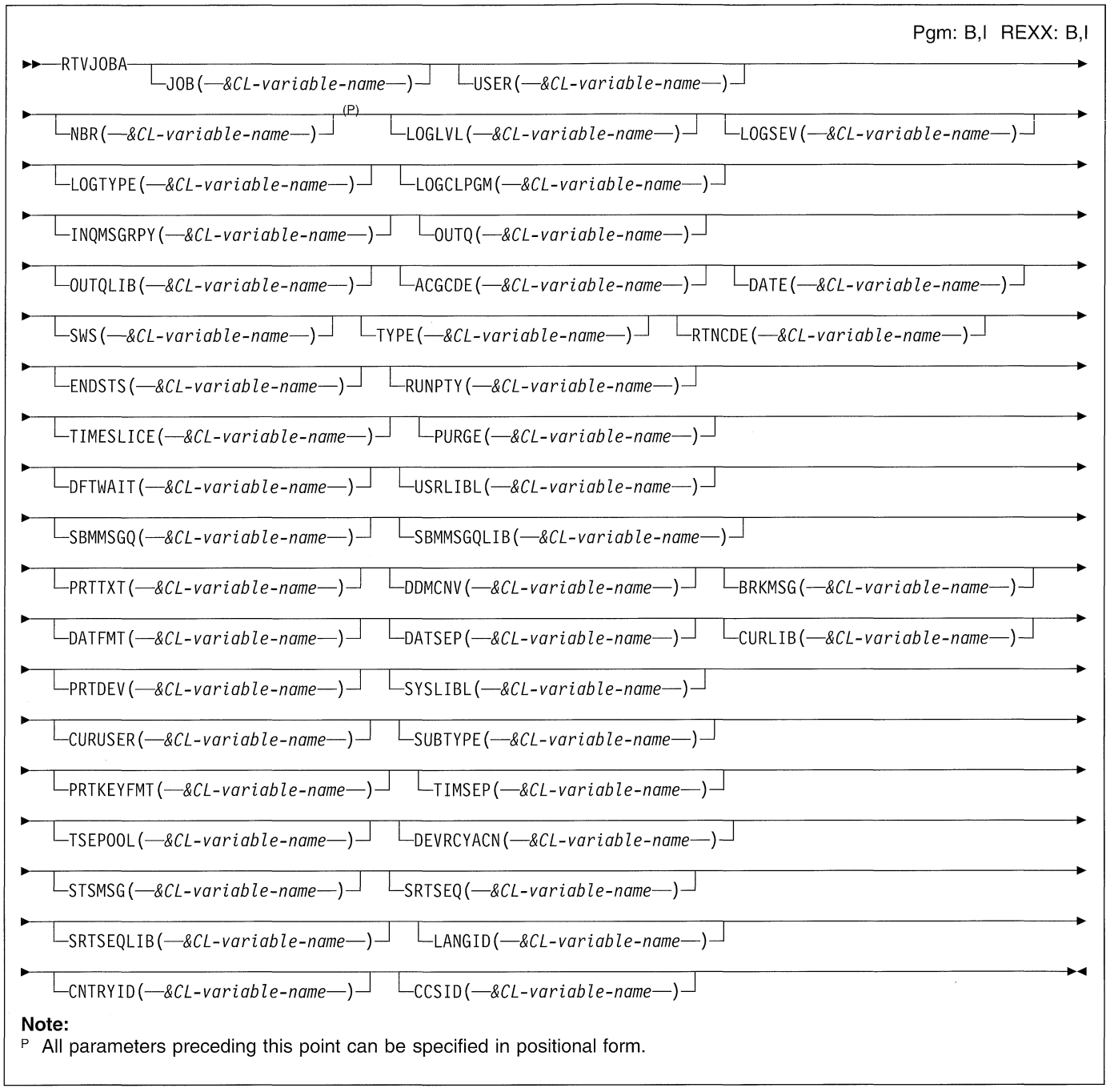
The contents of the CL variables returned are as follows:

```
&GRPJOBN: GROUPJ2
&GRPJOBL:

GROUPJ2  0002  50 characters of text for
               this group job...
GROUPJ1  0001  50 characters of text for
               this group job...
```

Fourteen more entries, full of blanks

```
&GRPCOUNT:      002
&MSGQNAME:      GROUPMSGQ
&MSGQLIB:       QGPL
&PRVJOB:        GROUPJ1      0001
&CTLCODE:       010
```

RTVJOBA (Retrieve Job Attributes) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Job Attributes (RTVJOBA) command is used in a control language (CL) program to retrieve the values of one or more job attributes and to place those values into the specified CL variable. The attributes are retrieved for the job in which this command is used. The following attributes can be retrieved:

- The job name (JOB), user profile name (USER), and job number (NBR)
- The message logging level (LOGLVL), severity level (LOGSEV), and text level (LOGTYPE)
- The CL program logging flag (LOGCLPGM) value
- The inquiry message handling (INQMSGRPY) value
- The output queue (OUTQ) being used for spooled output and the library (OUTQLIB) containing the output queue
- The accounting code (ACGCDE) for the job
- The date when the job was started (DATE)
- The values of the eight job switches (SWS) associated with the job
- The job type (TYPE), which can be batch or interactive

- The return code (RTNCDE) set by the last RPG, COBOL, DFU, or conversion reformat utility program that has completed processing in the job
- The end status (ENDSTS), which indicates whether a controlled end operation affecting the job is in progress
- The running (or processing) priority (RUNPTY) and the time slice (TIMESLICE) for the job's routing steps
- The value of the main storage purge flag (PURGE)
- The default wait time (DFTWAIT) for routing step processing
- The user portion of the job's library list (USRLIBL)
- The submitter specified job completion message queue (SBMMSGQ) and the library that contains it (SBMMSGQLIB)
- The current value of the print text (PRTTXT) job attribute
- The distributed data management (DDM) conversation setting (DDMCNV)
- The job's break message handling mode (BRKMSG)
- The date format (DATFMT) and date separator character (DATSEP) for the job
- The name of the current library for the job (CURLIB)
- The name of the printer device (PRTDEV)
- The system portion of the library list (SYSLIBL)
- The name of the current user profile (CURUSER)
- The subtype of a job environment (SUBTYPE)

The CL prompt for this command lists the minimum length for retrieved variables next to the appropriate parameters. For character variables, a single number is shown. For decimal variables, two numbers are shown. The first number indicates the minimum variable length and the second number indicates the minimum number of decimal positions.

Restriction: This command is valid only within a CL program.

Optional Parameters

JOB

Specifies, if the job name is returned, the name of the CL variable that receives the name of the job. The variable must be a character variable with a minimum length of 10 characters. If the job name has fewer characters than the variable allows, the value is padded on the right with blanks.

USER

Specifies, if the user name is returned, the name of the CL variable that receives the name of the user profile associated with the job when the job was started. The user name is the second part of the qualified job name. The variable must be a character variable with a minimum length of 10 characters. If the user name has fewer characters than the variable allows, the value is padded on the right with blanks.

NBR

Specifies, if the job number is being returned, the name of the CL variable that receives the unique 6-character number assigned to the job by the system. The job number is the first part of the qualified job name.

LOGLVL

Specifies the name of the CL variable that receives the 1-character value, ranging from 0 through 4, that is the message logging level used to determine the type of messages logged in the job log. The variable must be a character variable with a minimum length of 1 character.

LOGSEV

Specifies the name of the CL variable that receives the 2-digit value, ranging from 00 through 99, which is the minimum severity level a message must have before it is logged in the job log. The variable must be a 2-digit decimal variable specified with no decimal positions.

LOGTYPE

Specifies the name of the CL variable that receives the special value that indicates the level of text that appears for any message written to the job log. The variable must be a character variable with a minimum length of 10 characters. If the special value has fewer characters than the variable allows, the value is padded on the right with blanks.

LOGCLPGM

Specifies the name of the CL variable that receives the special value that indicates whether processed commands in a CL program are being logged in the job log. Note that this logging flag only has meaning for commands processed in CL programs which were created with LOG(*JOB) specified on the Create CL Program (CRTCLPGM) command. Additional information on the job's CL program logging flag is in the LOGCLPGM parameter description in the Change Job (CHGJOB) command. The variable must be a character variable with a minimum length of 10 characters. If the special value has fewer characters than the variable allows, the value is padded on the right with blanks.

INQMSGRPY

Specifies the name of the CL variable that receives the special value that indicates how inquiry messages are being handled by the job. Additional information on the value returned in the variable is in the INQMSGRPY parameter description in the Change Job (CHGJOB) command. The variable must be a character variable with a minimum length of 10 characters. If the special value has fewer characters than the variable allows, the value is padded on the right with blanks.

OUTQ

Specifies the name of the CL variable that receives the name of the output queue being used by the job for spooled output. The variable must be a character variable with a minimum length of 10 characters. If the output queue name has fewer characters than the variable allows, the value is padded on the right with blanks.

The special value *DEV can be retrieved. This value indicates that the output queue with the same name as the printer device, which is specified on either the printer file or the job description, will be used for the spooled output of this job.

OUTQLIB

Specifies the name of the CL variable that receives the name of the library that contains the output queue being used by the job for spooled output. The library name is the second part of the qualified output queue name (*output-queue/library-name*). The variable must be a character variable with a minimum length of 10 characters. If the output queue library name has fewer characters than the variable allows, the value is padded on the right with blanks.

ACGCDE

Specifies the name of the CL variable that receives the accounting code for the job. The variable must be a character variable with a minimum length of 15 characters. If the accounting code has fewer characters than the variable allows, the value is padded on the right with blanks.

DATE

Specifies, if the job date is to be returned, the name of the CL variable that receives the date assigned to the job by the system when the job was started. The variable must be a character variable with a minimum length of 6 characters. The job date is returned in the system date format specified by the system value QDATFMT.

SWS

Specifies, if the job switch settings are being returned, the name of the CL variable that receives the status (on or off) value of the eight job switches used by the job. The job switches are retrieved as a single 8-character value with each of the characters specifying a 1 (on) or 0 (off) as the value of the associated switch. The CL variable must be a character variable with a minimum length of 8 characters.

TYPE

Specifies, if the type of job environment is being returned, the name of the CL variable that receives the 1-character value representing the environment of the job. A character value of 0 indicates the job is running as a batch job, and a 1 indicates an interactive job. The variable must be a character variable with a minimum length of 1 character.

RTNCDE

Specifies (if the completion status of an RPG, COBOL, DFU, or conversion reformat utility program is being returned) the name of the CL variable that receives the 5-digit decimal return code. The return code is set by these programs before they return to the programs that called them. The return code indicates the completion status of the last program (of these types) that has completed processing within the job, as follows:

0 Normal return (RPG, COBOL, DFU, or Conversion Reformat Utility)

1 LR (last record) indicator on (RPG)

2 Error–no halt indicator set (RPG, COBOL, DFU, or Conversion Reformat Utility)

3 Halt indicator set on (one of the RPG indicators H1 through H9)

The CL variable must be a five-position decimal variable with no decimal positions.

ENDSTS

Specifies, if checking for a controlled end operation, the name of the CL variable that receives the end status. The single-character value indicates whether a controlled end that affects the job is currently being performed. A value of 1 indicates that the system, the subsystem in which the job is running, or the job itself is being ended; a value of 0 indicates no controlled end is being performed. The CL variable must be a character variable with a minimum length of 1 character.

RUNPTY

Specifies the name of the CL variable that receives the 2-digit value, ranging from 1 through 99, that is the run (or processing) priority for routing steps that are part of the job. For additional information on run priority, refer to this parameter description in the CHGJOB (Change Job) command. The variable must be a 2-digit decimal variable specified with no decimal positions.

TIMESLICE

Specifies the name of the CL variable that receives the 7-digit value, ranging from 1 through 9999999, that is the maximum number of milliseconds that a job's routing step can run when it is given processing time. For additional information on time slice, refer to this parameter description under the CHGJOB (Change Job) command. The variable must be a 7-digit decimal variable specified with no decimal positions.

PURGE

Specifies the name of the CL variable that receives the special value that indicates whether the job is marked as eligible to move from main storage at the end of a time slice or a long wait condition. For additional information on job purging, refer to this parameter description under the CHGJOB (Change Job) command. The variable must be a character variable with a minimum length of 10 characters. If the special value has fewer characters than the variable allows, the value is padded on the right with blanks.

DFTWAIT

Specifies the name of the CL variable that receives the 7-digit value, ranging from 1 through 9999999 (or -1 if the value is set to *NOMAX), which is the default for the maximum number of seconds that the system waits for a machine instruction to be completely processed. For additional information on wait-time, refer to the DFTWAIT parameter description under the CHGJOB (Change Job) command. The variable must be a 7-digit decimal variable specified with no decimal positions.

USRLIBL

Specifies the name of the CL variable that receives the user portion of the job's library list. The variable must be

a character variable with up to 275 variables. Each library name returned is left-justified in an 11-character field and padded on the right with blanks.

SBMMSGQ
Specifies the name of the CL variable that receives the name of a message queue. If this parameter is coded for a RTVJOBA command in a CL program which is part of a batch job, the value returned is associated with the MSGQ parameter on the JOB or SBMJOB command which caused the batch job to start running. If *NONE was specified for the MSGQ parameter of the JOB or SBMJOB commands, or the CL program that contains the RTVJOBA command is not part of a batch job, *NONE is returned to the CL variable coded for this parameter. The variable must be a character variable with a minimum length of 10 characters. If the message queue name has fewer characters than the variable allows, the value is padded on the right with blanks.

SBMMSGQLIB
Specifies the name of the CL variable that receives the name of the library that contains the message queue described above. (See the SBMMSGQ parameter description above.) The library name is the second part of the qualified message queue name. If *NONE is returned for the SBMMSGQ parameter, it is also returned for this parameter. The variable must be a character variable with a minimum length of 10 characters. If the message queue library name has fewer characters than the variable allows, the value is padded on the right with blanks.

```
/* Declare Variables */
DCL &LIBL *CHAR 275
DCL &CHGLIBL *CHAR 285
/*  save library list  */
RTVJOBA USRLIBL(&LIBL)
   •
   •
   •
/*  temporarily change library list  */
CHGLIBL LIBL(MYLIB QGPL)
   •
   •
   •
/*  build command string  */
CHGVAR &CHGLIBL ('CHGLIBL (' *CAT &LIBL *TCAT ')')

/*  restore library list  */
CALL QCMDEXC (&CHGLIBL 285)
```

The above commands retrieve the user portion of the library list so that later it can be restored from its temporary state, where only MYLIB and QGPL are in the user portion of the library list, to its original state. If there are no libraries on the user portion of the library list, then blanks are returned in the variable.

PRTTXT
Specifies the name of the CL variable that receives the print text for the job. This must be a character variable with a minimum length of 30 characters. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

DDMCNV
Specifies the name of a CL variable that receives the special value that indicates the action taken for DDM conversations on the job. The variable must be a character variable with a minimum length of 5 characters. If the special value has fewer characters than the variable allows, the value is padded on the right with blanks.

BRKMSG
Specifies the name of a CL variable that receives the special value that indicates the mode for break message handling that is in effect for the job. The variable must be a character variable with a minimum length of 7 characters. If the special value has fewer characters than the variable allows, the value is padded on the right with blanks.

DATFMT
Specifies the name of a CL variable that receives the special value being used as the date format for the job. The variable must be a character variable with a minimum length of 4 characters. If the special value has fewer than 4 characters, the value is padded on the right with blanks.

DATSEP
Specifies the name of a CL variable that receives the character being used as the date separator character for the job. The variable must be a character variable with a minimum length of 1 character.

CURLIB
Specifies the name of a CL variable that is used to retrieve the name of the current library for the job. The variable must be a character variable with a minimum length of 10 characters. If the current library name has fewer characters than the variable allows, the value is padded on the right with blanks.

Note: If the job does not have a current library, a value of *NONE is returned in this variable.

PRTDEV
Specifies the name of the CL variable that receives the name of the printer device. The variable must be a character variable with a minimum length of 10 characters. If the printer device name has fewer characters than the variable allows, the value is padded on the right with blanks.

SYSLIBL
Specifies the name of the CL variable that receives the system portion of the job's library list. The variable must be a character variable with up to 165 characters. Each library name returned is left-justified in an 11-character field and padded on the right with blanks.

CURUSER
Specifies, if the user name is returned, the 10-character CL variable that receives the name of the current user

profile. If the current user name has fewer characters than the variable allows, the value is padded on the right with blanks.

SUBTYPE
Specifies, if the subtype of job environment is being returned, the 1-character CL variable that receives the name of the job environment. The valid character values are:

Character	Value
*	Job has no subtype
E	Job is running as an evoked job
T	Job is running as a Multiple Requester Terminal (MRT) job
J	Job is running as a prestart job
P	Job is running as a print driver

PRTKEYFMT
Specifies the name of the CL variable that receives the print key format for the job. The variable must have a minimum length of 7 characters. The special value *NONE, *PRTBDR, *PRTHDR, or *PRTALL is returned.

TIMSEP
Specifies the name of a CL variable that receives the character being used as the time separator character for the job. The variable must be a character variable with a minimum length of one character.

TSEPOOL
Specifies the name of the CL variable that receives the special value indicating whether interactive jobs are moved to another main storage pool when they reach the time slice end. The variable must be a character variable with a minimum length of 10 characters.

DEVRCYACN
Specifies the name of the CL variable that receives the special value indicating the recovery action to take for the job when an I/O error is encountered on the *REQUESTER device for interactive jobs. The variable must be a character variable with a minimum length of 13 characters.

STSMSG
Specifies the name of the CL variable that receives the special value indicating how status messages are handled for the job. The variable must be a character variable with a minimum length of 7 characters.

SRTSEQ
Specifies the name of the CL variable that receives the name of the sort sequence table used for the job. The special value *LANGIDUNQ, *LANGIDSHR, or *HEX can be returned to the variable. The variable must be a character variable with a minimum length of 10 characters.

SRTSEQLIB
Specifies the name of the CL variable that receives the name of the library containing the sort sequence table to be used for the job. The variable must be a character variable with a minimum length of 10 characters. If SRTSEQ is *LANGIDUNQ, *LANGIDSHR, or *HEX, blanks are returned in the variable.

LANGID
Specifies the name of the CL variable that receives the value indicating the language identifier to be used for the job. The variable must be a character variable with a minimum length of 3 characters.

CNTRYID
Specifies the name of the CL variable that receives the value indicating the country identifier to be used for the job. The variable must be a character variable with a minimum length of 2 characters.

CCSID
Specifies the name of a CL variable that receives the CCSID value being used for the job. The variable must be a 5-digit decimal variable with no decimal positions.

Examples

```
RTVJOBA   NBR(&JOBNBR)  DATE(&JOBDATE)
```

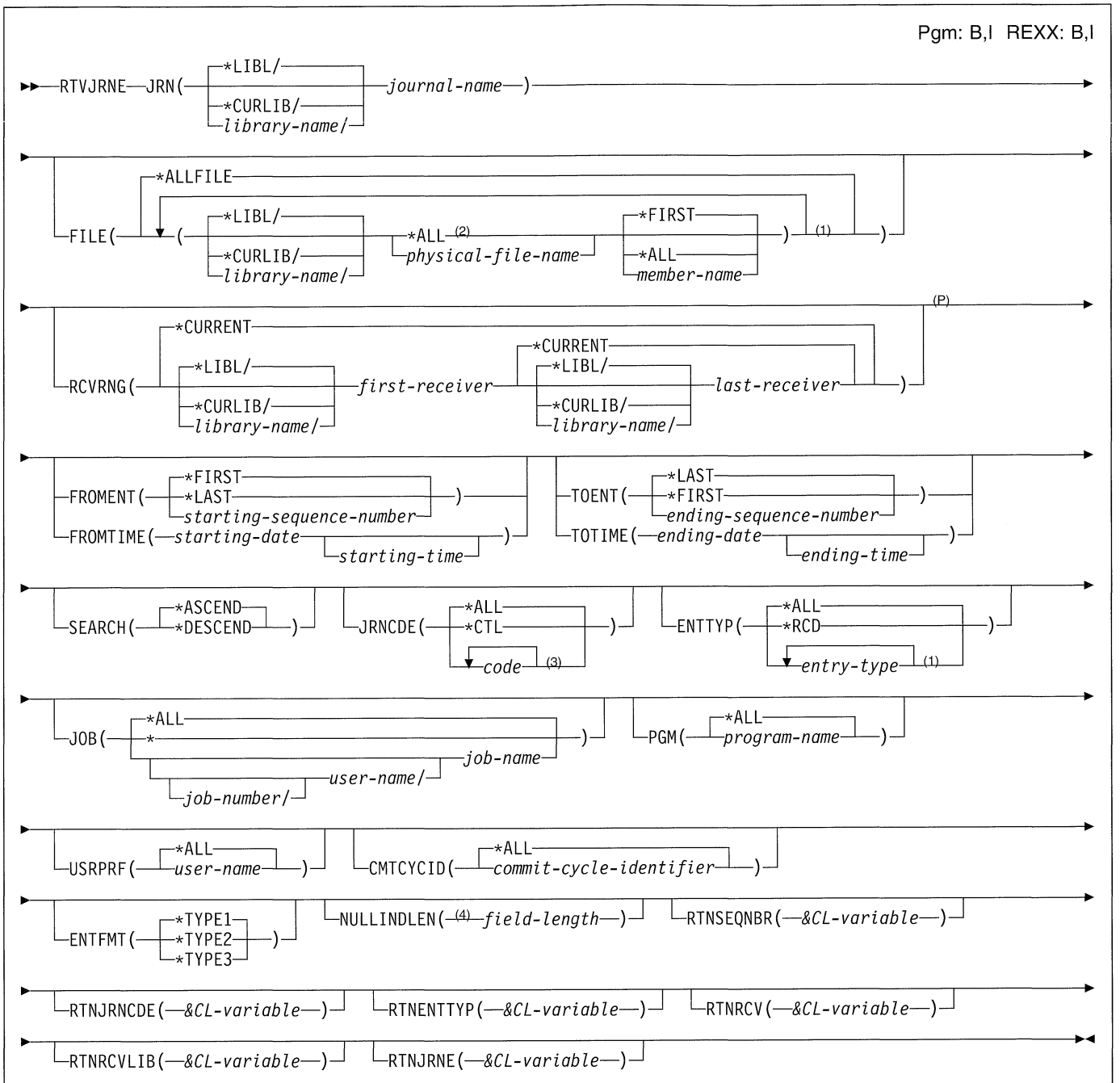
This command retrieves the job number and job date attributes from the job in which this command is located. The 6-digit job number is copied into the CL variable &JOBNBR. The job date is copied into the CL variable &JOBDATE; both variables must be 6 characters in length. The format of the date is determined by the contents of the system value QDATFMT, which controls the system date format.

```
/* Declare Variables */
DCL &LIBL *CHAR 275
DCL &CHGLIBL *CHAR 285

/*  save library list   */
RTVJOBA USRLIBL(&LIBL)
•
•
•
/*  temporarily change library list  */
CHGLIBL LIBL(MYLIB QGPL)
•
•
•
/*  build command string  */
CHGVAR &CHGLIBL ('CHGLIBL (' *CAT &LIBL *TCAT ')')

/*  restore library list  */
CALL QCMDEXC (&CHGLIBL 285)
```

The above command retrieves the user portion of the library list so that it later can be restored from its temporary state, where only MYLIB and QGPL were in the user portion of the library list, to its original state.

If there are no libraries on the user portion of the library list, blanks are returned in the variable. If a library on the library list has been deleted, the value '*DELETED' is put in the variable position for that name.

RTVJRNE (Retrieve Journal Entry) Command



Notes:

1 A maximum of 50 repetitions

2 If *ALL is specified instead of physical-file-name, the format must be library-name/*ALL.

P All parameters preceding this point can be specified in positional form.

3 A maximum of 10 repetitions.

4 The NULLINDLEN parameter can be specified only if ENTFMT(*TYPE3) is specified.

Purpose

The Retrieve Journal Entry (RTVJRNE) command allows the user to retrieve a journal entry and place the results in CL variables. The CL variables contain information, such as the sequence number of the retrieved entry, and are useful in automating certain types of recovery functions. The search for a journal entry can be restricted to a file, to a range of journal receivers, to a range of journal entries, to a journal code, to an entry type, to a job, to a program, to a user profile, or to a commit cycle identifier. Multiple limitation criteria can be specified. If more than one journal entry satisfies the search values specified, the first occurrence of a journal entry satisfying all of the specified search values is returned. If there is no journal entry satisfying the search values specified, the command ends with an escape message, and the return CL variables (RTNSEQNBR, RTNJRNCDE, RTNENTTYP, RTNRCV, RTNRCVLIB and RTNJRNE) remain the same.

The order of the search through the journal entries can be ascending or descending. The search order is determined by the value specified in the SEARCH parameter. The value for the FROM parameter must come before the value specified for the TO parameter in the specified search order.

The CL prompt for this command lists the minimum length for retrieved variables next to the correct parameters. For character variables, a single number is shown. For decimal variables, two numbers are shown. The first number indicates the minimum variable length, and the second number indicates the minimum number of decimal positions.

Restrictions:

1. If the sequence number is reset in the range of receivers specified, the first occurrence of either the FROMENT or TOENT parameter is used, if they are specified.
2. If more than one search value is specified, a journal entry being retrieved must satisfy all of the search values. Search values are FILE, JRNCDE, ENTTYP, JOB, PGM, and CMTCYCID. If none are specified, the first entry found that satisfies the values specified for RCVRNG, FROMENT, TOENT, FROMTIME, and TOTIME is retrieved.

Required Parameter

JRN

Specifies the qualified name of the journal from which the journal entry is retrieved.

The name of the journal can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

journal-name: Specify the name of the journal from which the journal entry is retrieved.

Optional Parameters

FILE

Specifies a maximum of 50 qualified file names whose journal entries are retrieved. The FILE parameter also specifies the name of the file member whose journal entries are to be retrieved.

If the specified physical file member currently exists on the system, all journal entries in the specified receiver range for that physical file member are retrieved. If the specified physical file member does not exist on the system, journal entries in the specified receiver range for any physical file member of that name that previously existed on the system are retrieved.

The single value *ALLFILE can be specified on the list of two values.

***ALLFILE:** The search for the entry being retrieved is not limited by a particular file name.

Element 1: Physical File Name

The name of the physical file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

***ALL:** The search for the entry being retrieved is for all files in the library that are currently being journaled. When FILE(*ALL) is specified, the file library name cannot be *LIBL.

physical-file-name: Specify the name of the physical database file for which a journal entry is retrieved.

Element 2: Member Name

***FIRST:** An entry for the first member in the file is retrieved.

***ALL:** Journal entries for currently existing members in the file are converted for output.

member-name: Specify the name of the member for which a journal entry is retrieved.

If the specified physical file does not exist on the system, specify either *ALL or a specific file member name.

RCVRNG

Specifies the qualified names of the first (beginning) and last (ending) journal receivers used in the search for a

journal entry to be retrieved. The system starts the search with the first journal receiver (as specified by the first value) and proceeds through the receiver chain until the last receiver (as specified by the second value) is processed. If dual receivers (pairs of receivers added or removed at the same time) are used at any time, the first of the receivers is used when chaining through the receivers. The Work with Journal Attributes (WRKJRNA) command can be used to display the order of the receivers in the receiver chain. If any problem is found in the receiver chain before the search operation begins, such as damaged or off-line receivers, the system attempts to use the second of the dual receivers. If the second of the receivers is damaged or off-line, or if a problem is found during the operation, the operation ends.

If SEARCH(*ASCEND) is specified, journal receivers must be specified in the order of oldest to newest. If SEARCH(*DESCEND) is specified, journal receivers must be specified in the order of newest to oldest.

Element 1: First Receiver Name

***CURRENT:** The journal entries in the currently attached receiver are searched for the specified journal entries to retrieve.

The name of the first journal receiver can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

first-receiver-name: Specify the name of the first journal receiver that contain journal entries to be retrieved.

Element 2: Last Receiver Name

***CURRENT:** The search for a journal entry continues for all journal receivers (in the chain that began with the receiver specified by the first parameter value) through the currently attached journal receiver.

The name of the last journal receiver can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

last-receiver-name: Specify the name of the last journal receiver containing journal entries to be searched. If the end of the receiver chain is reached before a receiver of this name is found, an error message is sent and no journal entry is retrieved.

Note: If the maximum number of receivers in the range is larger than 256, an error message is sent and no journal entry is retrieved.

FROMENT

Specifies the first journal entry considered for retrieval.

***FIRST:** The first journal entry in the specified journal receiver range is the first entry considered for retrieval. If SEARCH(*DESCEND) is specified, FROMENT(*FIRST) is valid only if TOENT(*FIRST) is also specified.

***LAST:** The last journal entry in the specified journal receiver range is the first entry considered for retrieval. If SEARCH(*ASCEND) is specified, FROMENT(*LAST) is valid only if TOENT(*LAST) is also specified.

starting-sequence-number: Specify the sequence number of the journal entry that is the first entry considered for retrieval.

FROMTIME

Specifies the date and time of the first journal entry considered for retrieval. The first journal entry found with the specified date and time or the next later journal entry is the starting point for the search.

starting-date: Specify the starting date using the format defined by the system values QDATFMT and, if separators are used, QDATSEP.

starting-time: Specify the starting time. The time is specified in 24-hour format with or without a time separator as follows:

- With a time separator, specify a string of 5 or 8 digits where the time separator separates the hours, minutes, and seconds. If this command is entered from the command line, the string must be enclosed in apostrophes. If a time separator other than the separator specified for your job is used, this command fails.
- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where **hh** = hours, **mm** = minutes, and **ss** = seconds. Valid values for **hh** range from 00 through 23. Valid values for **mm** and **ss** range from 00 through 59.

TOENT

Specifies the last journal entry considered for retrieval.

***LAST:** The search continues until the last journal entry in the specified journal receiver range is processed. If SEARCH(*DESCEND) is specified, TOENT(*LAST) is valid only if FROMENT(*LAST) is also specified.

***FIRST:** The search continues until the first journal entry in the specified journal receiver range is processed. If SEARCH(*ASCEND) is specified, TOENT(*FIRST) is only valid if FROMENT(*FIRST) is also specified.

ending-sequence-number: Specify the sequence number of the journal entry that is the final entry considered for retrieval.

Note: The values specified for the FROMENT and TOENT parameter can be the same (for example, FROMENT(234) and TOENT(234) can be specified).

TOTIME

Specifies the date and time of the last entry considered for retrieval. The first journal entry found with the specified date and time, or the latest earlier journal entry is the ending point for the search.

ending-date: Specify the ending date using the format defined by the system values QDATFMT and, if separators are used, QDATSEP.

ending-time: Specify the ending time. See the FROMTIME parameter for a description of time formats.

SEARCH

Specifies the order in which the journal entries are searched to retrieve an entry.

***ASCEND:** The journal entries are searched in ascending order (from the oldest entry to the newest entry).

***DESCEND:** The journal entries are searched in descending order (from the newest entry to the oldest entry).

JRNCDE

Specifies the journal codes of the journal entries being considered for retrieval.

***ALL:** The search for the entry to retrieve is not limited to a specified journal code.

***CTL:** The entries considered for retrieval are the journal entries created to control the journal functions. The journal codes for these are "J" and "F".

code: Specify the journal code that limits the search for the journal entries to retrieve. Only journal entries that contain the journal code are considered for retrieval. A list of journal codes that can be specified is in the *Advanced Backup and Recovery Guide*.

ENTTYP

Specifies the entry type of the entries being considered for retrieval.

***ALL:** The search for the entry to retrieve is not limited to a specified entry type.

***RCD:** Only entries that have an entry type for record-level operations are converted. The following entry types are valid: BR, DL, DR, PT, PX, UB, UP, and UR.

entry-type: Specify the entry type that limits the search for the entry journal entries to retrieve. Only journal entries that contain the specified entry type are considered for retrieval. Up to 50 valid entry types can be specified. A list of valid entry types is in the *Advanced Backup and Recovery Guide*.

JOB

Specifies whether the journal entries being retrieved are limited to the journal entries for a specified job. Only journal entries for the specified job are considered for retrieval. If no job name is given, all of the journal entries that contain the simple name of the job are considered for retrieval.

A job identifier is a special value or a qualified name with up to three elements. For example:

```
*ALL
*
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ALL:** The search is not limited to entries for a specified job.

***:** The search is limited to entries for the current job. Only entries for the requesting job are considered for retrieval.

job-name: Specify the name of the job whose journal entries are being considered for retrieval.

user-name: Specify the name of the user of the job whose journal entries are being considered for retrieval.

job-number: Specify the number of the job whose journal entries are being considered for retrieval.

PGM

Specifies the name of a program whose journal entries are considered for retrieval.

***ALL:** The search is not limited to entries for a specified program.

program-name: Specify the name of the program whose journal entries are considered for retrieval. Only journal entries for this program are considered for retrieval.

USRPRF

Specifies that the journal entries to be considered for conversion are limited to the journal entries for a specified user profile name.

***ALL:** The retrieval is not limited to entries for a specified user profile.

user-name: Specify the name of the user profile whose journal entries are to be considered for retrieval. Only changes for this user profile that were journaled are considered for retrieval.

CMTCYCID

Specifies that the journal entries considered for retrieval are limited to the journal entries that contain the specified commit cycle identifier.

***ALL:** The search is not limited to entries for a specified commit cycle identifier.

commit-cycle-identifier: Specify the commit cycle identifier that limits the search for the entry to retrieve. Only journal entries that contain the commit cycle identifier are considered for retrieval.

ENTFMT

Specifies the format of the journal entries being retrieved.

***TYPE1:** The retrieved journal entries do not include the user profile field, the system name field, or the null value indicators.

***TYPE2:** The retrieved entries include the information returned when ENTFMT(*TYPE1) is specified, the user profile field giving the name of the user who logged the displayed journal entries, and the name of the system on which the entry was sent.

***TYPE3:** The retrieved journal entries include the information returned when ENTFMT(*TYPE2) is specified, and the null value indicators.

For lists containing detailed information on the format of the retrieved journal entries, see the RTNJRNE parameter.

NULLINDLEN

Specifies the length, in bytes, used for the null value indicators portion of the retrieved entry. This parameter is valid only if ENTFMT(*TYPE3) is specified. Valid values range from 1 to 8000.

If the retrieved journal entry has fewer null value indicators than the specified field length, the trailing bytes in the null value indicators field will be set to 'F0'X.

RTNSEQNBR

Specifies the name of the CL decimal variable in the program into which the journal entry sequence number of the retrieved journal entry is copied. If a CL variable name is not specified, the journal entry sequence number is not copied into the program. The specified variable must be a decimal variable that has a length of ten positions with no decimal positions.

RTNJRNCDE

Specifies the name of the CL character variable in the program into which the journal code of the retrieved journal entry is copied. If a CL variable name is not specified, the journal code of the retrieved journal entry is not copied into the program. The specified variable must be a character variable with a minimum length of 1 character. If the length of the variable is longer than 1 character, it is padded on the right with blanks.

RTNENTTYP

Specifies the name of the CL character variable in the program into which the entry type of the retrieved journal entry is copied. If a CL variable name is not specified, the entry type of the retrieved journal entry is not copied into the program. The specified variable must be a character variable with a minimum length of 2 characters. If the length of the variable is longer than 2 characters, it is padded on the right with blanks.

RTNRCV

Specifies the name of the CL character variable in the program into which the journal receiver name from where the returned journal entry was retrieved is copied. If the CL variable name is not specified, the journal receiver name is not copied into the program. The specified variable must be a character variable with a minimum length of 10 characters. If the length of the variable is longer than 10 characters, it is padded on the right with blanks.

RTNRCVLIB

Specifies the name of the CL character variable in the program into which journal receiver library name identified by the returned journal entry was retrieved is copied. If the CL variable name is not specified, the journal receiver library name is not copied into the program. The specified variable must be a character variable with a minimum length of 10 characters. If the length of the variable is longer than 10 characters, it is padded on the right with blanks.

RTNJRNE

Specifies the name of the CL character variable in the program into which the retrieved journal entry is copied. If a CL variable name is not specified, the retrieved journal entry is not copied into the program. The specified variable must be a character variable. If the retrieved journal entry is longer than the variable's field length, the entry is truncated. If the entry is shorter, it is padded on the right with blanks.

The journal entry can be retrieved in one of three possible formats. If ENTFMT(*TYPE1) is specified, then the format of the fields in the retrieved entry is as follows:

Field Name	Field Attributes
ENTRY LENGTH	TYPE(*DEC) LEN(5 0)
SEQUENCE NUMBER	TYPE(*DEC) LEN(10 0)
JOURNAL CODE	TYPE(*CHAR) LEN(1)
JOURNAL ENTRY TYPE	TYPE(*CHAR) LEN(2)
DATE	TYPE(*CHAR) LEN(6)
TIME	TYPE(*DEC) LEN(6 0)
JOB NAME	TYPE(*CHAR) LEN(10)
USER NAME	TYPE(*CHAR) LEN(10)
JOB NUMBER	TYPE(*DEC) LEN(6 0)
PROGRAM NAME	TYPE(*CHAR) LEN(10)
OBJECT NAME	TYPE(*CHAR) LEN(10)
OBJECT LIBRARY	TYPE(*CHAR) LEN(10)
MEMBER NAME	TYPE(*CHAR) LEN(10)
COUNT/RRN	TYPE(*DEC) LEN(10 0)
FLAG	TYPE(*CHAR) LEN(1)
COMMIT CYCLE ID	TYPE(*DEC) LEN(10 0)
RESERVED	TYPE(*CHAR) LEN(8)
ENTRY-SPECIFIC DATA	TYPE(*CHAR) LEN(up to 9844)

If ENTFMT(*TYPE2) is specified, the format of the fields in the retrieved entry is as follows:

Field Name	Field Attributes
ENTRY LENGTH	TYPE(*DEC) LEN(5 0)
SEQUENCE NUMBER	TYPE(*DEC) LEN(10 0)
JOURNAL CODE	TYPE(*CHAR) LEN(1)
JOURNAL ENTRY TYPE	TYPE(*CHAR) LEN(2)
DATE	TYPE(*CHAR) LEN(6)
TIME	TYPE(*DEC) LEN(6 0)
JOB NAME	TYPE(*CHAR) LEN(10)
USER NAME	TYPE(*CHAR) LEN(10)
JOB NUMBER	TYPE(*DEC) LEN(6 0)
PROGRAM NAME	TYPE(*CHAR) LEN(10)
OBJECT NAME	TYPE(*CHAR) LEN(10)
OBJECT LIBRARY	TYPE(*CHAR) LEN(10)
MEMBER NAME	TYPE(*CHAR) LEN(10)
COUNT/RRN	TYPE(*DEC) LEN(10 0)
FLAG	TYPE(*CHAR) LEN(1)
COMMIT CYCLE ID	TYPE(*DEC) LEN(10 0)
USER PROFILE	TYPE(*CHAR) LEN(10)
SYSTEM NAME	TYPE(*CHAR) LEN(8)
RESERVED	TYPE(*CHAR) LEN(20)
ENTRY-SPECIFIC DATA	TYPE(*CHAR) LEN(up to 9844)

If ENTFMT(*TYPE3) is specified and a value is specified on the NULLINDLEN parameter, the format of the retrieved journal entry is as follows:

Field Name	Field Attributes
ENTRY LENGTH	TYPE(*DEC) LEN(5 0)
SEQUENCE NUMBER	TYPE(*DEC) LEN(10 0)
JOURNAL CODE	TYPE(*CHAR) LEN(1)
JOURNAL ENTRY TYPE	TYPE(*CHAR) LEN(2)
TIMESTAMP	TYPE(*TIMESTAMP) LEN(26)
JOB NAME	TYPE(*CHAR) LEN(10)
USER NAME	TYPE(*CHAR) LEN(10)
JOB NUMBER	TYPE(*DEC) LEN(6 0)
PROGRAM NAME	TYPE(*CHAR) LEN(10)
OBJECT NAME	TYPE(*CHAR) LEN(10)
OBJECT LIBRARY	TYPE(*CHAR) LEN(10)
MEMBER NAME	TYPE(*CHAR) LEN(10)
COUNT/RRN	TYPE(*DEC) LEN(10 0)
FLAG	TYPE(*CHAR) LEN(1)
COMMIT CYCLE ID	TYPE(*DEC) LEN(10)
USER PROFILE	TYPE(*CHAR) LEN(10)
SYSTEM NAME	TYPE(*CHAR) LEN(8)
NULL VALUE INDICATORS	TYPE(*CHAR) field-length*
ENTRY-SPECIFIC DATA	TYPE(*CHAR) ((up to 9850) minus (field length))**

* The length of this field is the length specified on the NULLINDLEN parameter.

** The length of this portion of the entry depends on the length specified on the RTNJRNE parameter and the length specified on the NULLINDLEN parameter.

Examples

Example 1

Assume the following variables are specified:

```
DCL &SEQ        TYPE(*DEC) LEN(10 0)
DCL &JRNENT     TYPE(*CHAR) LEN(200)
DCL &RCVNAME    TYPE(*CHAR) LEN(10)
DCL &RCVLIB     TYPE(*CHAR) LEN(10)
```

and this command is run:

```
RTVJRNE   JRN(MYLIB/JRNA)  ENTTYP(PR)
  RTNSEQNBR(&SEQ#)  RTNJRNE(&JRNENT)
```

Since no starting journal entry is specified in this command, the first entry from the currently-attached receiver MYLIB/JRNA is considered for retrieval. The first entry in any receiver is always an identifier for the previously-attached receiver. This first receiver entry is known as a type PR entry, and it contains the name of the previously attached receiver in its entry-specific data. The PR entry is the first entry in ascending order in the currently attached receiver; when it is found, the entry is placed into a CL variable named &JRNENT.

Then Change Variable (CHGVAR) can be used to separate the name and library of the previous journal receiver, found in the entry specific data, as follows:

```
CHGVAR &RCVNAME  (%SST(&JRNENT 126 10))
CHGVAR &RCVLIB   (%SST(&JRNENT 136 10))
```

Example 2

Assume the following variables are specified:

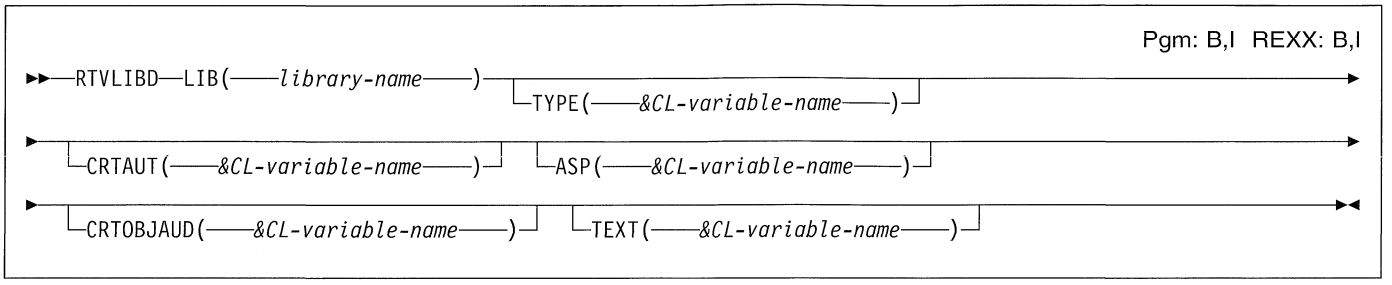
```
DCL &ENTNO        TYPE(*DEC) LEN(10 0)
DCL &JCODE        TYPE(*CHAR) LEN(1)
DCL &ETYPE        TYPE(*CHAR) LEN(2)
DCL &RCVNAME      TYPE(*CHAR) LEN(10)
DCL &RCVLIB       TYPE(*CHAR) LEN(10)
DCL &JENTRY       TYPE(*CHAR) LEN(205)
```

and this command is run:

```
RTVJRNE   JRN(MYLIB/JRNLA)  FILE(LIB1/A MBR3)
  RCVRNG(RCVLIB/RCV30 RCVLIB/RCV27)
  ORDER(*DESCEND)  JRNCDE(R)  ENTTYP(UP DL)
  JOB(000666/QPGMR/PRESTRT)  PGM(WAKEUP)
  USRPRF(MAC7)  ENTFMT(*TYPE2)
  RTNSEQNBR(&ENTNO)  RTNJRNCDE(&JCODE)
  RTNENTTYP(&ETYPE)  RTNRCV(&RCVNAME)
  RTNRCVLIB(&RCVLIB)  RTNJRNE(&JENTRY)
```

This command gets a journal entry, searching in descending order the journal receiver chain from receiver RCV30 in library RCVLIB to receiver RCV27 in library RCVLIB, journaled through journal JRNLA in library MYLIB, and copies the entry into the specified CL variables. The retrieved entry is an UPDATE or DELETE entry with journal code R from member MBR3 in file A in library LIB1, created in job 000666/QPGMR/PRESTRT in program WAKEUP by user profile MAC7. The retrieved journal entry includes the user profile field. The sequence number of the retrieved entry is copied into CL variable &ENTNO. The journal code of the retrieved entry is copied into CL variable &JCODE. The entry type of the retrieved entry is copied into CL variable &ETYPE. The name of the journal receiver from which the returned entry was retrieved is copied into &RCVNAME. The library name of the journal receiver from which the returned entry was retrieved is copied into &RCVLIB.

RTVLIBD (Retrieve Library Description) Command



Purpose

The Retrieve Library Description (RTVLIBD) command is used to retrieve the description of a library.

The CL prompt for this command lists the minimum length for each variable next to the parameters being retrieved. For character variables, a single number is shown. For decimal variables, two numbers are shown. The first number indicates the minimum variable length and the second number indicates the minimum number of decimal positions.

Restriction: The user must have authority to the library from which the attributes are being retrieved.

Required Parameter

LIB

Specifies the name of the library for which information is being retrieved. If a variable is specified, it must be 10 characters in length and include a library name.

Optional Parameters

TYPE

Specifies the name of a 10-character CL variable that is used to retrieve the type of the library.

The value that can be returned is PROD or TEST.

CRTAUT

Specifies the name of a 10-character CL variable that is used to return the create authority value of the library.

The value that can be returned is *SYSVAL, *CHANGE, *ALL, *USE, *EXCLUDE, or the name of an authorization list.

ASP

Specifies the name of a decimal variable that is used to retrieve the identifier of the auxiliary storage pool (ASP) from which the system allocates storage for the library. In CL programs, this must be a decimal variable length of (2 0). The following values can be returned:

1	The object is in the system auxiliary storage pool.
2-16	The object is in a user auxiliary storage pool.

CRTOBJAUD

Specifies the name of a 10-character variable used to return the auditing value of the library. The values that can be retrieved include *SYSVAL, *NONE, *USRPRF, *CHANGE, and *ALL. See the CRTOBJAUD parameter on the Create Library (CRTLIB) command for more information.

TEXT

Specifies the name of a 50-character CL variable that is used to return the text description to a library.

Example

```
CRTLIB LIB(TESTLIB) CRTAUT(*ALL)
  TEXT('John Smith library')
```

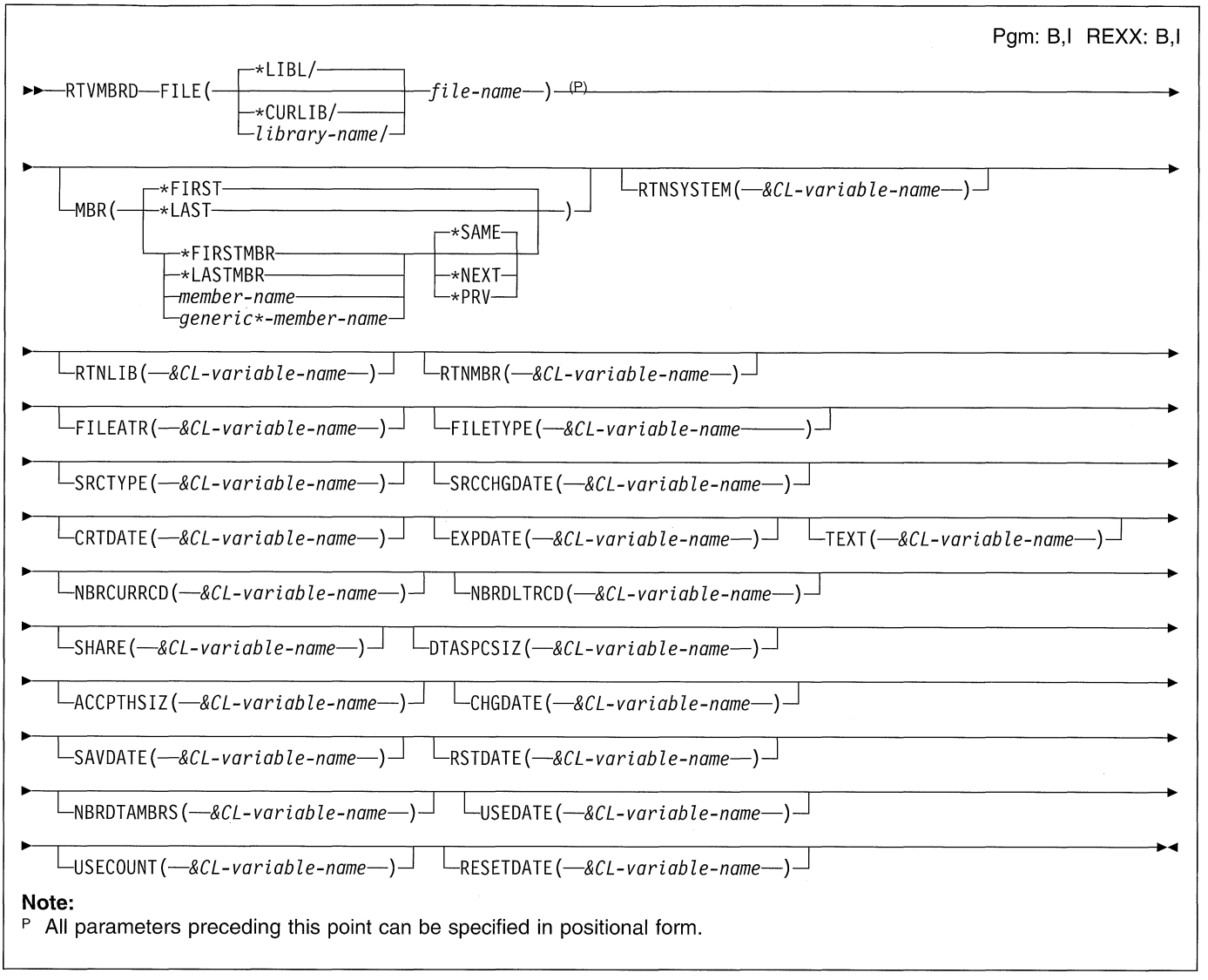
This command creates John Smith's library.

```
DCL &CRTAUT *CHAR10
RTVLIBD LIB(TESTLIB)
  CRTAUT(&CRTAUT)
```

When user Smith calls a CL program containing the above, the following values are returned:

```
&CRTAUT  *ALL
```

RTVMBRD (Retrieve Member Description) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Member Description (RTVMBRD) command is used in a CL program to retrieve (return) the member-level information (in CL variables) from a database file.

The values are returned (copied) to the specified CL variables in the program. The following kinds of member information can be retrieved:

- The library name.
- The member name.
- The file attribute.
- The file type.
- The source type.
- The source date.
- The date created.
- The expiration date.
- The member text.
- The number of nondeleted records.
- The number of deleted records.
- The open data path status (shared or not shared).
- The data space size.
- The access path size.
- The date changed.
- The date saved.
- The date restored.
- The number of data members.
- The last date used.
- The days count used.
- The date and days count was reset.

Required Parameters

FILE

Specifies the qualified name of the file that contains the member description being retrieved.

Note: *OBJOPR authority to the file and *READ authority to the library are required for the member description being retrieved.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the file.

Optional Parameters

MBR

Specifies the member whose description is retrieved.

***FIRST:** The first member in the database file is used.

***LAST:** The last member of the specified physical file is retrieved.

Element 1: Member to Retrieve

***FIRSTMBR:** The first member in a list sorted by name is retrieved.

***LASTMBR:** The last member in a list sorted by name is retrieved.

member-name: Specify the name of the reference member. The reference member is a base from which the *NEXT, *PRV, or *SAME member is retrieved.

generic-member-name:* Specify the generic member name being retrieved. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be retrieved only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Element 2: Operation to Perform on Member

***SAME:** The value does not change.

***NEXT:** The member immediately after the reference member is retrieved.

***PRV:** The member immediately before the reference member is retrieved.

RTNSYSTEM

Specifies the name of a 4-character CL variable used to retrieve an indication of the system from which the description is retrieved. The possible values are *LCL (file found on the local system) and *RMT (file found on remote system).

RTNLIB

Specifies the name of a 10-character CL variable used to retrieve the name of the library where the specified file is located.

RTNMBR

Specifies the name of a 10-character CL variable used to retrieve the name of the member whose description is being retrieved.

FILEATR

Specifies the name of a 3-character CL variable used to retrieve the file attribute. The possible values are *PF (retrieves the attribute for a physical file member) and *LF (retrieves the attribute for a logical file member).

FILETYPE

Specifies the name of a 5-character CL variable used to retrieve the file type. The possible values are *DATA (retrieves the file type for a data member) and *SRC (retrieves the file type for a source member).

SRCTYPE

Specifies the name of a 10-character CL variable used to retrieve the source member type (if this is a source member). Blanks are returned if this is not a source member.

SRCCHGDATE

Specifies the name of a 13-character CL variable used to retrieve the century, date, and time the last source file member was changed. The format is CYYMMDDHHMMSS where C=Century (0=1940 through 1999 and 1=2000 through 2039), Y=Year, M=Month, D=Day, H=Hour, M=Minutes, and S=Seconds. Blanks are returned if no date is available or if the date of remote non-AS/400 system or non-System/38 files are retrieved.

CRTDATE

Specifies the name of a 13-character CL variable used to retrieve the member creation century, date, and time. The format is CYYMMDDHHMMSS where C=Century (0=1940 through 1999 and 1=2000 through 2039), Y=Year, M=Month, D=Day, H=Hour, M=Minutes, and S=Seconds.

EXPDATE

Specifies the name of a 7-character CL variable used to retrieve the member expiration century, date, and time. The format is CYYMMDD where C=Century (0=1940 through 1999 and 1=2000 through 2039), Y=Year, M=Month, and D=Day. *NONE is returned if no date is available.

TEXT
Specifies the name of a 50-character CL variable used to retrieve the text description of the file member. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

NBRCURRCD
Specifies the name of a 10-position decimal CL variable used to retrieve the current number of nondeleted records in the member. If the member is a logical member, the number of index entries is returned.

NBRDLTRCD
Specifies the name of a 10-position decimal CL variable used to retrieve the current number of deleted records in this member. Zero (0) is returned for keyed logical files.

SHARE
Specifies whether the open data path (ODP) for the member description is shared with other programs in the routing step. When an ODP is shared, the programs accessing the file share facilities such as the file status and the buffer.

More information on shared database files is in the *Database Guide*.

DTASPCSIZ
Specifies the name of a 15-position decimal CL variable used to retrieve the data space size (in bytes) of this member. Zero (0) is returned if this is a logical member.

ACCPTHSIZ
Specifies the name of a 12-position decimal CL variable used to retrieve the access path size (in bytes) for this member. Zero (0) is returned if the member is non-keyed. Remote non-AS/400 system and non-System 38 files return a value of zero (0).

CHGDATE
Specifies the name of a 13-character CL variable used to retrieve the member change century, date, and time. The format is CYYMMDDHHMMSS where C=Century (0=1940 through 1999 and 1=2000 through 2039), Y=Year, M=Month, D=Day, H=Hour, M=Minutes, and S=Seconds.

SAVDATE
Specifies the name of a 13-character CL variable used to retrieve the member save century, date, and time. The format is CYYMMDDHHMMSS where C=Century (0=1940 through 1999 and 1=2000 through 2039), Y=Year, M=Month, D=Day, H=Hour, M=Minutes, and S=Seconds. Blanks are returned if no date is available. Remote non-AS/400 system and non-System 38 files return blanks.

RSTDATE
Specifies the name of a 13-character CL variable used to retrieve the member restore century, date, and time. The format is CYYMMDDHHMMSS where C=Century (0=1940 through 1999 and 1=2000 through 2039), Y=Year, M=Month, D=Day, H=Hour, M=Minutes, and S=Seconds. Blanks are returned if there is no date available.

NBRDTAMBRS
Specifies the name of a 2-position decimal CL variable used to retrieve the number of data members for this logical member. If the member is a physical member, zero (0) is returned.

USEDATE
Specifies the name of a 7-character CL variable that is used to retrieve the date the object was used last. The date will be returned in the form CYYMMDD where C=Century (0=1940 through 1999 and 1=2000 through 2039), Y=Year, M=Month, and D=Day. If the object does not have a last-used date, blanks will be returned.

USECOUNT
Specifies the name of a 5 position decimal CL variable used to return the number of days the object has been used. If the object does not have a last used date, 0 will be returned.

RESETDATE
Specifies the name of a 7-character CL variable that is used to return the date the days-used count was last reset to 0. The date will be returned in the form CYYMMDD where C=Century (0=1940 through 1999 and 1=2000 through 2039), Y=Year, M=Month, and D=Day. If the days used count has not been reset, blanks will be returned.

Examples

Assume the user has a file named MYFILE in library MYLIB (which is the current library) with members QMEMBER, BMEMBER, ZMEMBER, and JMEMBER (created in that order).

Also assume the following variables are specified in the CL program:

```
DCL  &LIB        TYPE(*CHAR) LEN(10)
DCL  &MBR        TYPE(*CHAR) LEN(10)
DCL  &SYS        TYPE(*CHAR) LEN(4)
DCL  &MTYPE      TYPE(*CHAR) LEN(5)
DCL  &CRTDATE    TYPE(*CHAR) LEN(13)
DCL  &CHGDATE    TYPE(*CHAR) LEN(13)
DCL  &TEXT       TYPE(*CHAR) LEN(50)
DCL  &NBRRCD     TYPE(*DEC) LEN(10 0)
DCL  &SIZE       TYPE(*DEC) LEN(10 0)
```

This command is run:

```
RTVMBRD  FILE(*CURLIB/MYFILE)  MBR(BMEMBER *SAME)
  RTNLIB(&LIB)  RTNSYSTEM(&SYS)  RTNMBR(&MBR)
  FILEATR(&MTYPE)  CRTDATE(&CRTDATE)  TEXT(&TEXT)
  NBRCURRCD(&NBRRCD)  DTASPCSIZ(&SIZE)
```

The requested information is placed in the CL variables as follows:

- The current library name (MYLIB) is placed in the CL variable named &LIB.
- The system on which MYFILE was found is placed in the CL variable named &SYS. (*LCL means the file was found on the local system, and *RMT means the file was found on a remote system.)
- The member name (BMEMBER) is placed in the CL variable named &MBR.
- The file attribute of MYFILE is placed in the CL variable named &MTYPE. (*DATA means the member is a data member, and *SRC means the file is a source member.)
- The creation date of BMEMBER is placed in the CL variable named &CRTDATE.
- The text associated with BMEMBER is placed in the CL variable called &TEXT.
- The current number of records in BMEMBER is placed in the CL variable called &NBRRCD.
- The size of BMEMBER's data space (in bytes) is placed in the CL variable called &SIZE.

This command is run next:

```
RTVMBRD  FILE(&LIB/MYFILE)  MBR(&MBR *NEXT)
  RTNMBR(&MBR)  CRTDATE(&CRTDATE)  TEXT(&TEXT)
  NBRCURRCD(&NBRRCD)  DTASPCSIZ(&SIZE)
```

Then the requested information is placed in the CL variables as follows:

- The next members name after BMEMBER (JMEMBER since the file is searched in name order) in MYFILE is placed in the CL variable named &MBR.
- The creation date of JMEMBER is placed in the CL variable named &CRTDATE.
- The text associated with JMEMBER is placed in the CL variable called &TEXT.
- The current number of records in JMEMBER is placed in the CL variable called &NBRRCD.
- The size of JMEMBER's data space (in bytes) is placed in the CL variable called &SIZE.

The file can also be searched backwards. An example is:

```
RTVMBRD  FILE(*CURLIB/MYFILE)  MBR(ZMEMBER *PRV)
  RTNMBR(&MBR)  CHGDATE(&CHGDATE)  TEXT(&TEXT)
```

The requested information is placed in the CL variables as follows:

- The member name (QMEMBER since it is the member just previous to ZMEMBER in a name ordered list) is placed in the CL variable named &MBR.
- The date QMEMBER was last changed is placed in the CL variable named &CHGDATE.
- The text associated with QMEMBER is placed in the CL variable called &TEXT.

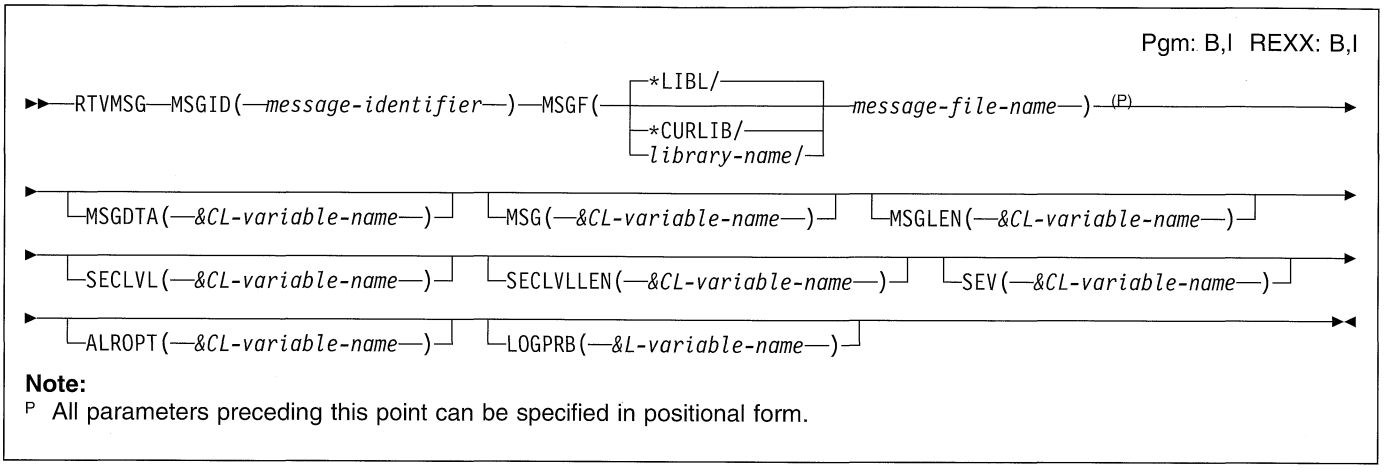
If only the first part of the member name is known, the user can do a Generic* (or partial name) search of the list of members, as follows:

```
RTVMBRD FILE(*LIBL/MYFILE) MBR(JM*) RTNMBR(&MBR)
  CHGDATE(&CHGDATE) TEXT(&TEXT)
```

The requested information is placed in the CL variables as follows:

- The member name (JMEMBER since it is the first member starting with the characters JM in a name ordered list) is placed in the CL variable named &MBR.
- The date JMEMBER was last changed is placed in the CL variable named &CHGDATE.
- The text associated with JMEMBER is placed in the CL variable called &TEXT.

RTVMSG (Retrieve Message) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Message (RTVMSG) command is used by a program to retrieve a specified predefined message from a message file and to copy it into CL variables in the program. Substitution values can be specified in the MSGDTA parameter (as a single character string containing one or more concatenated message data fields) to replace the substitution variables in the predefined message text. The program can later write the message to an output device file to be printed, for example.

The CL prompt for this command lists the minimum length for retrieved variables next to the parameters that have a minimum length. For character variables, a single number is shown. For decimal variables, two numbers are shown. The first number indicates the minimum variable length and the second number indicates the minimum number of decimal positions.

Restrictions: (1) This command is valid only in compiled CL programs. (2) The user of this command must have *USE authority for the message file and the library in which the message file is stored.

Required Parameters

MSGID

Specifies the message identifier of the predefined message being retrieved from the specified message file.

MSGF

Specifies the qualified name of the message file that contains the predefined message being retrieved.

The name of the message file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-file-name: Specify the name of the message file to use.

Optional Parameters

MSGDTA

Specifies the substitution values that are used in the retrieved message if the predefined message contains substitution variables. Either a character string or a CL variable containing the character string can be specified. As many substitution values can be specified in the character string as there are substitution variables in the predefined message. The values, which take the place of the substitution variables defined in the message text when the message was defined, must be specified by the following rules:

- Multiple values must be concatenated to form a single character string. The length of the entire character string of concatenated substitution values cannot exceed 512 characters. The *CL Programmer's Guide* has more details.
- Multiple values must be specified in the same order in the character string as the substitution variables were defined in the FMT parameter of the Add Message Description (ADDMSGD) command.
- Each value must be specified as long as its associated variable was defined, and the specified character string must be the same length as the sum of the message data fields defined in the message description.
- For multiple values, each substitution value can be specified as a CL program constant or CL variable; that is, any combination of constants and variables

can be specified if they are first concatenated into one character string and they are in the same format and sequence expected.

- The length of the substitution value should be the same length as the length defined for the substitution variables. If the substitution value length is longer than the substitution variable length, the message data is truncated. If the substitution value is shorter, it becomes a null field.

For more information on coding this parameter, refer to the description of the MSGDTA parameter in the SNDUSRMSG or SNDPGMMSG commands.

MSG
Specifies the name of the CL character variable in the program into which the first-level message text from the retrieved message is copied. If a CL variable name is not specified, the text is not copied into the program.

The specified variable must be a character variable. If the retrieved text is longer than the variable's field length, the text is truncated. If the text is shorter, it is padded to the right with blanks. Although this is a variable-length field, most first-level message text is designed to be less than 132 characters.

MSGLEN
Specifies the name of the CL decimal variable in the program into which the total length of the text available to be retrieved is copied. The length specified is the total length after the substitution values (specified in the MSGDTA parameter) are placed in the text.

The specified variable must be a decimal variable that has a length of five digits. If a CL variable name is not specified, the length is not copied into the program.

SECLVL
Specifies the name of the CL character variable in the program into which the second-level message text of the retrieved message is copied. If a variable name is not specified, the second-level message text is not copied into the program.

If the retrieved second-level message text is longer than the variable's field length, the text is truncated. If the text is shorter, it is padded on the right with blanks. Although this is a variable length field, most second-level message text is designed to be less than 3000 characters.

SECLVLLEN
Specifies the name of the CL decimal variable in the program into which the total length of the second-level message text being retrieved is copied. The length specified is the total length after the substitution values (specified in the MSGDTA parameter) are placed in the second-level message text.

The specified variable must be a decimal variable that has a length of five positions. If a CL variable name is not specified, the length is not copied into the program.

SEV
Specifies the name of the CL decimal variable into which the severity code of the retrieved message is copied. The specified variable must be a decimal variable that has a length of two positions. If a variable name is not specified, the severity code of the retrieved message is not copied into the program.

ALROPT
Specifies the name of the CL variable into which the alert option of the retrieved message is copied. The variable must be a character variable nine positions long.

LOGPRB
Specifies the name of the variable into which a Y (yes) or N (no) is returned indicating whether or not the message can be logged in the service activity log. The variable must be a character variable one position long.

Examples

Example 1: Replacing Substitution Variables

```
RTVMSG  MSGID(UIN0145)  MSGF(INVN)  MSG(&WORK)
  MSGDTA('any old time')
```

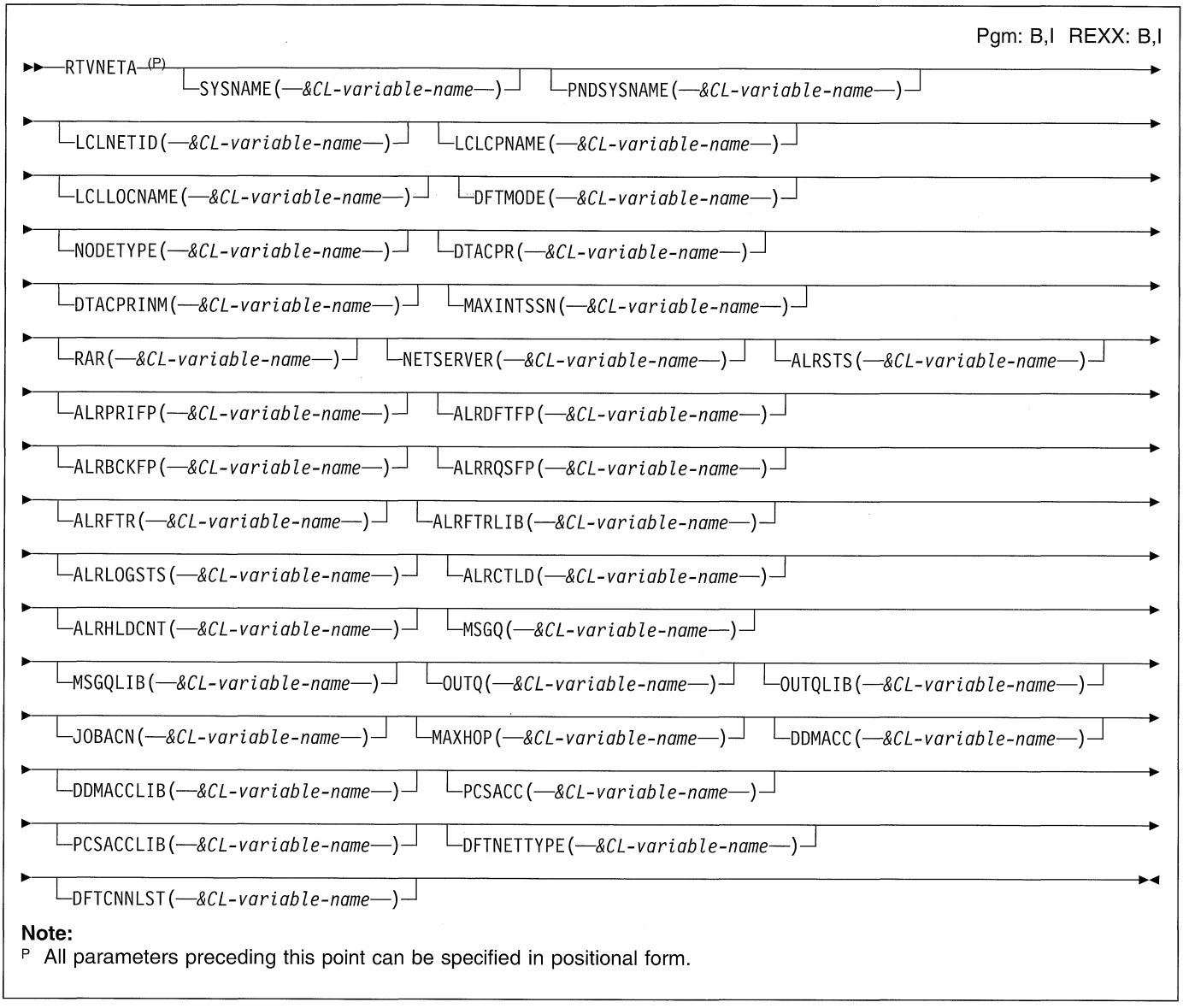
This command retrieves the message text of the message UIN0145 stored in the INVN message file. The retrieved text is copied into the CL variable &WORK after the substitution variables are replaced with the values *any*, *old*, and *time*. This example assumes that the substitution variables &1, &2, and &3 have been defined in the message as character variables, each 4 characters long.

Example 2: Retrieving First- and Second-Level Message Text

```
RTVMSG  MSGID(UIN0150)  MSGF(INV)  MSG(&MSG)
  SECLVL(&SECLVL)
```

This command retrieves the first- and second-level message text of the message UIN0150, which is stored in message file INV, and moves it into the CL variables &MSG and &SECLVL.

RTVNETA (Retrieve Network Attributes) Command



Note:

(P) All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Network Attributes (RTVNETA) command is used in a CL program to retrieve the network attributes of the system.

The values are returned (copied) to the specified CL variables in the program.

Restrictions:

1. This command is valid only in a compiled CL program.
2. The attributes of the network attribute and the receiving CL variable must be compatible.

Optional Parameters

SYSNAME

Specifies the name of the CL variable that receives the current system name. Specify the name of a character variable with a minimum length of 8 characters. If the system name has fewer characters than the variable allows, the value is padded on the right with blanks.

PNDSYSNAME

Specifies the name of the CL variable that receives the pending system name. Specify the name of the character variable with a minimum length of 8 characters. The value returned is blank if no pending system name is provided. If the system name has fewer characters than the variable allows, the value is padded on the right with blanks.

LCLNETID
Specifies the name of the CL variable that receives the current local network ID. Specify the name of the character variable with a minimum length of 8 characters. If the network ID has fewer characters than the variable allows, the value is padded on the right with blanks.

LCLCPNAME
Specifies the name of the CL variable that receives the current local control point. Specify the name of the character variable with a minimum length of 8 characters. If the control point has fewer characters than the variable allows, the value is padded on the right with blanks.

LCLLOCNAME
Specifies the local location name.

DFTMODE
Specifies the name of the CL variable that receives the default mode name. Specify the name of the character variable with a minimum length of 8 characters. If the default mode has fewer characters than the variable allows, the value is padded on the right with blanks.

NODETYPE
Specifies the name of the CL variable that receives the current APPN nodetype. Specify the name of the character variable with a minimum length of 8 characters.

The values that can be returned in the variable as the node type are:

***ENDNODE**
The node does not provide network services to other nodes but may participate in the APPN network by using the services of an attached network server or may operate in a peer-to-peer environment similar to migration end nodes.

***NETNODE**
The node provides intermediate routing, route selection services, and distributed directory services for local users and to end nodes and migration end nodes that it is serving.

DTACPR
Specifies the name of the CL variable that receives the current level of data compression. Specify the name of the decimal variable with a minimum length of 10 digits without decimal positions.

The values that can be returned in the variable as the data compression level are:

0 *NONE: Data compression is not allowed on the session.

-1 *REQUEST: Data compression is requested on the session by the local system. However, the request can be refused or changed to a lower compression level by the remote system. Data compression is allowed on the session if requested by the remote system.

-2 *ALLOW: Data compression is allowed on the session by the local system if requested by a remote system. The local system does not request compression.

-3 *REQUIRE: Data compression is required on the session. If the remote system does not change the levels of compression to the local system's exact requested levels, the session is not established. The data compression levels that the local system requires are the specified levels.

DTACPRINM
Specifies the name of the CL variable that receives the current level of intermediate node data compression. Specify the name of the decimal variable with a minimum length of 10 digits without decimal positions.

The values that can be returned in the variable as the intermediate node data compression levels are:

0 *NONE: The remote systems are not notified of a need to compress data when the AS/400 system is an SNA intermediate node.

-1 *REQUEST: The remote systems are requested to compress data when the AS/400 system is an SNA intermediate node.

MAXINTSSN
Specifies the name of the CL variable that receives the maximum number of intermediate sessions. Specify the name of the decimal variable with a minimum length of 5 digits without decimal positions.

RAR
Specifies the name of the CL variable that receives the route addition resistance. Specify the name of the decimal variable with a minimum length of 5 digits without decimal positions.

NETSERVER
Specifies the name of the CL variable that receives the list of network node servers. Specify the name of a character variable with a minimum length of 85 characters. If the server control point name or network ID has fewer characters than the variable allows, the value is padded on the right with blanks. The list can contain five node servers. Each server has the form: Network ID (9 characters) followed by the server name (8 characters). There are no separators. The network ID may contain the value *LCLNETID which specifies that the current network ID is used. If less than five node servers are specified, the remaining ones contain hexadecimal zeros for a name. As soon as the first null name is encountered in the list, it is safe to assume that the remaining names will also be null.

ALRSTS
Specifies the name of the CL variable that receives the current system alert status. Specify the name of a character variable with a minimum length of 10 characters. If

the alert status value has fewer characters than the variable allows, the value is padded on the right with blanks.

The values that can be returned in the variable as the current system alert status are:

***ON** Alert processing is currently active.

***UNATTEND** Alert processing is currently active, but the system is unattended.

***OFF** Alert processing is currently inactive.

ALRPRIFP

Specifies the name of the CL variable that receives the primary alert focal point. Specify the name of a character variable with a minimum length of 10 characters. If the primary focal point value has fewer characters than the variable allows, the value is padded on the right with blanks.

The values that can be returned in the variable as the primary alert focal point are:

***NO** The system is not an alert primary focal point.

***YES** The system is an alert primary focal point.

ALRDFTFP

Specifies the name of the CL variable that receives the default alert focal point. Specify the name of a character variable with a minimum length of 10 characters. If the default focal point value has fewer characters than the variable allows, the value is padded on the right with blanks.

The values that can be returned in the variable as the default alert focal point are:

***NO** The system is not an alert default focal point.

***YES** The system is an alert default focal point.

ALRBCKFP

Specifies the name of the CL variable that receives the name of the system that provides alert focal point services if the primary focal point is unavailable. You must specify the name of a character variable with a minimum length of 10 characters. If the back up system name has fewer characters than the variable allows, the value is padded on the right with blanks.

ALRRQSFP

Specifies the name of the CL variable that receives the name of the system that is requested to provide alert focal point services. You must specify the name of a character variable with a minimum length of 10 characters. If the requesting system name has fewer characters than the variable allows, the value is padded on the right with blanks.

ALRFTR

Specifies the name of the CL variable that receives the name of the active alert filter. You must specify the name of a character variable with a minimum length of 10 characters. If the alert filter name has fewer characters than the variable allows, the value is padded on the right with blanks.

ALRFTRLIB

Specifies the name of the CL variable that receives the name of the library that contains the alert filter definition. Specify the name of a character variable with a minimum length of 10 characters. If the library name has fewer characters than the variable allows, the value is padded on the right with blanks.

ALRLOGSTS

Specifies the name of the CL variable that receives the current alert logging status. Specify the name of a character variable with a minimum length of 10 characters.

The values that can be returned in the variable as the current alert logging status are:

***NONE** Do not log alerts.

***LOCAL** Log only locally generated alerts.

***RCV** Log only alerts received from other nodes.

***ALL** Log both locally generated and incoming alerts.

ALRCTLD

Specifies the name of the CL variable that receives the name of the controller through which alert messages are sent to another system when alert processing is active. Specify the name of a character variable with a minimum length of 10 characters. If the alert controller name has fewer characters than the variable allows, the value is padded on the right with blanks.

The values that can be returned in the variable are:

***NONE** There is no controller for alerts.

controller-name:
Specifies the name of the controller being used for alerts in an alert controller session. This controller is ignored if the system has a primary or default alert focal point (if, for example, the node is in another system's sphere of control).

ALRHLDCNT

Specifies the name of the CL variable that receives the maximum number of alerts that are created before the alerts are sent over the alert controller session (ALRCTLD network attribute). The alerts are held (queued) by the system until the specified number of alerts have been created. This parameter can be used to manage alerts that are sent over a limited resource by reducing the number of times alerts are sent. Specify the name of a decimal variable with a total length of 5 digits without decimal positions. The maximum number of alerts that can be created before the alerts are sent is 32,767.

Note: The ALRHLDCNT network attribute only applies when the ALRCTLD network attribute is used.

When management services sessions, APPN, and sphere of control support are used, the ALRHLDCNT value is ignored.

The value that can be returned in the variable as the maximum number of alerts is:

***NOMAX** The alerts are held indefinitely. The current alert hold count is the maximum value. The alerts can be sent at a later time by changing the ALRHLDCNT value to a lower value.

MSGQ
Specifies the name of the CL variable that receives the group message queue name. This must be a character variable with a minimum length of 10 characters. If the message queue name has fewer characters than the variable allows, the value is padded on the right with blanks.

MSGQLIB
Specifies the name of the CL variable that receives the name of the library that contains the system-default network message queue. Specify the name of a character variable with a minimum length of 10 characters. If the library name has fewer characters than the variable allows, the value is padded on the right with blanks.

OUTQ
Specifies the name of the CL variable that receives the system default network output queue name. Specify the name of a character variable with a minimum length of 10 characters. If the output queue name has fewer characters than the variable allows, the value is padded on the right with blanks.

OUTQLIB
Specifies the name of the CL variable that receives the name of the library that contains the system-default network message queue. Specify the name of a character variable with a minimum length of 10 characters. If the library name has fewer characters than the variable allows, the value is padded on the right with blanks.

JOBACN
Specifies the name of the CL variable that receives the current job action for input streams received through the network. Specify the name of a character variable with a minimum length of 10 characters. If the job action value has fewer characters than the variable allows, the value is padded on the right with blanks.

The values that can be returned in the variable as the current job action are:

***FILE** Input streams received from the network are filed in the queue of network files for the user to which it was sent.

***REJECT** Input streams received from the network are rejected.

***SEARCH** The table of network job entries is searched to determine the action for any input streams received.

MAXHOP
Specifies the name of the CL variable that receives the maximum number of systems in the SNADS network so that a distribution queue originating at this node may be received and rerouted on the path to its final destination. Specify the name of a decimal variable with a total length of 5 digits without decimal positions.

DDMACC
Specifies the name of the CL variable that receives the current system action for DDM requests from other systems. Specify the name of a character variable with a minimum length of 10 characters. If the DDM access value has fewer characters than the variable allows, the value is padded on the right with blanks.

The values that can be returned in the variable as the current system action for DDM requests are:

***REJECT** The system does not allow any DDM requests from remote systems. However, the system may use DDM to access files on remote systems. (A remote system cannot access any file on a system that specifies *REJECT).

***OBJAUT** All file requests are allowed and controlled by the object authorizations on the system (normal system object level security).

program-name
Specifies the name of the customer validation program that can supplement object level security. This user-exit program can restrict user access to *PUBLIC and private files. The target DDM support calls the user program for each reference to a file. The user-exit program indicates to DDM if the request should proceed or end.

DDMACCLIB
Specifies the name of the CL variable that receives the name of the library that contains the DDM access program. Specify the name of a character variable with a minimum length of 10 characters. If the library name has fewer characters than the variable allows, the value is padded on the right with blanks. If *REJECT or *OBJAUT is returned for the DDMACC parameter, the value for DDMACCLIB is all blanks.

PCSACC
Specifies the name of the CL variable that receives the current system action for PCS requests. Specify the name of a character variable with a minimum length of 10 characters. If the PCS access value has fewer characters than the variable allows, the value is padded on the right with blanks.

The values that can be returned in the variable as the current system action for PCS requests are:

***REJECT** The system does not allow any PCS requests.

***OBJAUT** All PCS requests are allowed and controlled by the object authorizations on the system.

program-name
Specifies the name of the customer supplied PC Support host system application exit program that can supplement system object level security.

PCSACCLIB
Specifies the name of the CL variable that receives the name of the library that contains the PCS access program. Specify the name of a character variable with a minimum length of 10 characters. If the library name has fewer characters than the variable allows, the value is padded on the right with blanks. If *REJECT or *OBJAUT is returned for the PCSACC parameter, the value for PCSACCLIB is all blanks.

DFTNETTYPE
Specifies the name of the CL variable that receives the system default value for the integrated services digital network (ISDN) network type. The user must specify a character variable name with a minimum length of 10 characters.

The values that can be returned in the variable as the network type are:

***ATTG3:** Indicates an ISDN in the United States or Canada that uses AT&T 5ESS version G3 switching equipment.

***ATT5E42:** Indicates an ISDN in the United States or Canada that uses AT&T 5ESS version 5E4.2 switching equipment.

***ATT5E5:** Indicates an ISDN in the United States or Canada that uses AT&T 5ESS version 5E5 switching equipment.

***ATT5E6:** Indicates an ISDN in the United States or Canada that uses AT&T 5ESS version 5E6 switching equipment.

***BTNR191:** Indicates an ISDN in the United Kingdom controlled by British Telecomm.

***CCITT88:** The ISDN default values recommended by the International Telegraph and Telephone Consultative Committee (CCITT) in 1988 are used.

***DBP1TR6:** Indicates an ISDN controlled by Germany's PTT. (Deutsche Bundes Poste 1TR6).

***ETSI:** Indicates an ISDN that uses the European Telecommunications Standards Institute (ETSI, also known as EuroISDN) standard.

***FTVN2:** Indicates version 2 of the ISDN controlled by France's post telephone and telegraph administration (PTT). (France Telecom Numeris VN2).

***INSNET64:** Indicates the INSNET64 ISDN controlled by Japan's Nippon Telephone and Telegraph Public Corporation (NTT).

***NISDN:** Indicates an ISDN that uses the National ISDN-1 or National ISDN-2 standard for North America.

***NT100B29:** Indicates an ISDN in the United States or Canada that uses Northern Telecom DMS100 Version BCS-29 switching equipment.

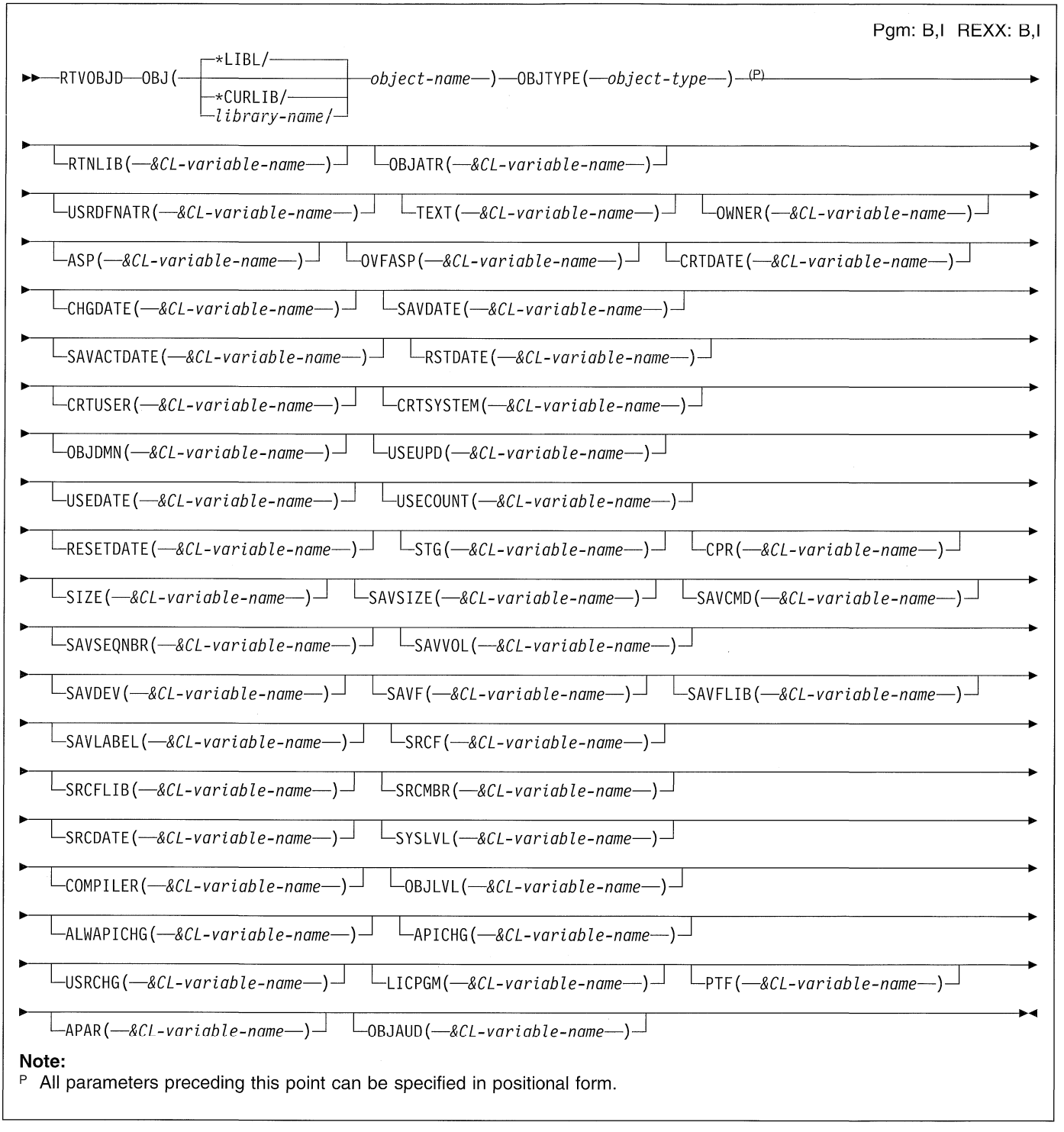
DFTCNNLST
Specifies the name of the CL variable that receives the system default value for the ISDN connection list. The user must specify a character variable name with a minimum length of 10 characters.

Example

```
DCL  VAR(&SNAME)  TYPE(*CHAR)  LEN(8)
DCL  VAR(&HOPS)  TYPE(*DEC)  LEN(5 0)
RTVNETA  SYSNAME(&SNAME)  MAXHOP(&HOPS)
```

This command retrieves the current system name and the maximum number of systems in a SNADS network so that a distribution queue entry can be received and rerouted on the path to its destination.

RTVOBJD (Retrieve Object Description) Command



Note:

(P) All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Object Description (RTVOBJD) command returns the description of a specific object to a CL program or a REXX procedure.

Required Parameters

OBJ

Specifies the qualified name of the object being retrieved. If no library name is given, *LIBL is used to find the specified object.

The name of the object being retrieved can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

object-name: Specify the name of the object that is being retrieved.

OBJTYPE
Specify the type of object being retrieved. If a variable is specified, it can be up to 8 characters in length. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Optional Parameters

RTNLIB
Specifies the name of a variable used to return the name of the library that contains the object. In a CL program, the variable returned has a length of 10 characters. If *LIBL or *CURLIB is specified for the library name on the OBJ parameter, the value returned is the name of the library where the object was found.

OBJATR
Specifies the name of a variable used to return an extended attribute of the object, such as a program or file type. In a CL program the variable returned has a length of 10 characters. For example, the variable may be returned with PROD (production) or CLP (control language program). No asterisk (*) precedes the value.

USRDFNATR
Specifies the name of a variable used to return the user-defined attribute of an object. In a CL program the variable returned has a length of 10 characters. Blanks are returned if the retrieved object does not have a user-defined attribute.

TEXT
Specifies text that briefly describes the object being retrieved. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

OWNER
Specifies the name of a variable that is used to return the name of the object owner's user profile. In a CL program, the variable returned has a length of 10 characters.

ASP
Specifies the name of a variable that is used to return the auxiliary storage pool ID. In a CL program, the variable returned has a decimal length of (2 0). The following values can be returned:

1 The object is in the system auxiliary storage pool.

2-16 The object is in a user auxiliary storage pool.

OVFASP
Specifies the name of a variable that is used to return the Object Overflowed ASP flag. A value of 1 is returned if the object overflowed the ASP in which it resides. A value of 0 is returned if the object does not overflow the ASP. It is not possible for an object residing in the system ASP to overflow its ASP, therefore, a value of 0 is always returned for objects in the system ASP.

CRTDATE
Specifies the name of a variable used to return the date and time the object was created. In a CL program, the variable returned has a length of 13 characters. The value is returned in the form CYYMMDDHHMMSS, where C = century; '0' indicates the 20th century and '1' indicates the 21st century; YY = year, MM = month, DD = day, HH = hour, MM = minutes, and SS = seconds.

CHGDATE
Specifies the name of a variable used to return the date and time the object was last changed. In a CL program, the variable returned has a length of 13 characters. The variable is returned in the same format as CRTDATE or is returned blank if the object has not been changed.

SAVDATE
Specifies the name of a variable used to return the date and time the object was last saved. In a CL program, the variable returned has a length of 13 characters. The variable is returned in the same format as CRTDATE or is returned blank if the object has not been saved.

SAVACTDATE
Specifies the name of a variable used to return the date and time the object was last saved. In a CL program, the variable returned has a length of 13 characters. Blanks are returned if the object has not been saved or if SAVACT(*NO) was specified on the last save operation for the object.

RSTDATE
Specifies the name of a variable used to return the date and time the object was last restored. In a CL program, the variable returned has a length of 13 characters. The variable is returned in the same format as CRTDATE or is returned blank if the object has not been restored.

CRTUSER
Specifies the name of a variable used to return the name of the user that created the object. In a CL program, the variable returned has a length of 10 characters.

CRTSYSTEM
Specifies the name of a variable used to return the name of the system on which the object was created. In

a CL program, the variable returned has a length of 8 characters.

OBJDMN
Specifies the name of a CL variable used to return the object domain value for an object. In a CL program, the variable returned has a length of 2 characters. A value of *U indicates the object is in the user domain; a value of *S indicates the object is in the system domain.

USEUPD
Specifies the name of a variable that indicates whether the object usage information is updated for this object type. In a CL program, the variable returned has a length of 1 character. The indicator is returned as 'Y' or 'N'. If 'N' is returned, the last used date for the object is blank.

USEDATE
Specifies the name of a 7-character CL variable used to return the date the object was used last. In a CL program, the variable returned has a length of 7 characters. The date is returned in the form CYYMMDD. If the object does not have a last used date, blanks are returned.

USECOUNT
Specifies the name of a variable used to return the number of days the object has been used. In a CL program, the variable returned has a decimal length of (5 0). If the object does not have a last used date, zero (0) is returned.

RESETDATE
Specifies the name of a variable used to return the date on which the days used count was reset to 0. In a CL program, the variable returned has a length of 7 characters. The date is returned in the form CYYMMDD. If the days used count has not been reset, blanks are returned.

STG
Specifies the name of a variable used to return the storage status of the object data. In a CL program, the variable returned has a length of 10 characters. *FREE is returned if the object data has been freed and the object has been suspended. *KEEP is returned if the object data has not been freed and the object has not been suspended.

CPR
Specifies the name of a variable used to return the compression status of the object. In a CL program, the variable returned has a length of 1 character. (Y) is returned if the object is compressed. (N) is returned if the object is permanently decompressed. (T) is returned if the object is temporarily decompressed. (F) is returned if the object is eligible for compression but is saved with storage freed. (X) is returned if the object is ineligible for compression.

SIZE
Specifies the name of a variable used to return the size of the object in bytes. In a CL program, the variable returned has a decimal length of (15 0).

SAVSIZE
Specifies the name of a variable used to return the size of the object in bytes of storage at the time of the last save operation. In a CL program, the variable returned has a decimal length of (15 0). The variable is returned with zeros if the object has not been saved.

SAVCMD
Specifies the name of a variable used to return the command used to save the object. In a CL program, the variable returned has a length of 10 characters. The variable is returned blank if the object has not been saved.

SAVSEQNBR
Specifies the name of a variable used to return the tape sequence number generated when the object was saved on tape. In a CL program, the variable returned has a decimal length of (4 0). The variable is returned with zeros if the object has not been saved.

SAVVOL
Specifies the name of a variable used to return the tape or diskette volumes used for saving the object. In a CL program, the variable returned has a length of 71 characters. The variable returns up to 10 six-character volumes. Each volume ID entry is separated by a single blank. The volume IDs begin in character positions 1, 8, 15, 22, 29, 36, 43, 50, 57, and 64. If more than 10 volumes are used, a '1' is returned in the 71st character of the variable, otherwise the 71st character is blank. The variable is returned blank if the object was saved to a save file the last time it was saved, or if it was never saved.

SAVDEV
Specifies the name of a variable used to return the type of device to which the object was last saved. In a CL program, the variable returned has a length of 10 characters. The variable is returned with *SAVF if the last save operation was to a save file. The variable is returned with *DKT if the last save operation was to a diskette. The variable is returned with *TAP if the last save operation was to tape. The variable is returned blank if the object was not saved.

SAVF
Specifies the name of a variable used to return the name of the save file if the object was saved to a save file. In a CL program, the variable returned has a length of 10 characters. The variable is returned blank if the object was not saved to a save file.

SAVFLIB
Specifies the name of a variable used to return the name of the library that contains the save file if the object was saved to a save file. In a CL program, the

variable returned has a length of 10 characters. The variable is returned blank if the object was not saved to a save file.

SAVLABEL

Specifies the name of a variable used to return the file label used when the object was saved. In a CL program, the variable returned has a length of 17 characters. The variable is returned blank if the object was not saved to tape or diskette. The value of the return variable corresponds to value specified for the LABEL parameter on the command used to save the object.

SRCF

Specifies the name of a variable used to return the name of the source file used to create the object. In a CL program, the variable returned has a length of 10 characters. The variable is returned blank if no source file was used to create the object.

SRCFLIB

Specifies the name of a variable used to return the name of the library that contains the source file that was used to create the object. In a CL program, the variable returned has a length of 10 characters. The variable is returned blank if no source file was used to create the object.

SRCMBR

Specifies the name of a variable used to return the name of the member in the source file (parameter SRCF). In a CL program, the variable returned has a length of 10 characters. The variable is returned blank if no source file was used to create the object.

SRCDATE

Specifies the name of a variable used to return the date and time the member in the source file was last updated. In a CL program, the variable returned has a length of 13 characters. The variable is returned in the same format as CRTDATE or is returned blank if the member is not updated.

SYSLVL

Specifies the name of a variable used to return the level of the operating system when the object was created. In a CL program, the variable returned has a length of 9 characters:

- A 3-character version level starting in character position 1
- A 3-character release level starting in character position 4
- A 3-character modification level starting in character position 7

The first character of the version level is always the letter 'V'; the first character of the release level is always the letter 'R'; the first character of the modification level is always the letter 'M'.

COMPILER

Specifies the name of a variable used to return the licensed program identifier, version level, release level, and modification level of the compiler. In a CL program, the variable returned has a length of 16 characters. The variable is returned with the 7-character program identifier starting in character position 1, the 3-character version level in character position 8, the 3-character release level in character position 11, and the 3-character modification level in character position 14. The first character of the version level is always the letter 'V'; the first character of the release level is always the letter 'R'; the first character of the modification level is always the letter 'M'. The variable is returned blank if no compiler was used.

OBJLVL

Specifies the name of a variable used to return the object control level for the created object. In a CL program, the variable returned has a length of 8 characters.

ALWAPICHG

Specifies the name of a variable used to return the Allow Change by Program flag. A '1' is returned if the object can be changed with the Change Object Description API, QLICOBJD. A '0' is returned if the object cannot be changed with the API.

APICHG

Specifies the name of a variable used to return the Changed by Program flag. A '1' is returned if the object has been modified with the Change Object Description API, QLICOBJD. A '0' is returned if the object has not been modified by the API.

USRCHG

Specifies the name of a variable used to return whether the program was modified by the user. In a CL program, the variable returned has a length of 1 character. A '1' is returned if the object was modified by the user. A '0' is returned if the object was not modified by the user.

LICPGM

Specifies the name of a variable used to return the name, version level, release level, and modification level of the licensed program if the retrieved object is part of a licensed program. In a CL program, the variable returned has a length of 16 characters:

- The 7-character name starting in character position 1
- The 3-character version level in character position 8
- The 3-character release level in character position 11
- The 3-character modification level in character position 14

The first character of the version level is always the letter 'V'; the first character of the release level is always the letter 'R'; the first character of the modification level

is always the letter 'M'. The variable is returned blank if the retrieved object is not a licensed program.

PTF
Specifies the name of a variable used to return the Program Temporary Fix number that resulted in the creation of the retrieved object. In a CL program, the variable returned has a length of 10 characters. The variable is returned blank for user created-objects.

APAR
Specifies the name of a variable used to return the Authorized Program Analysis Report identification. In a CL program, the variable returned has a length of 10 characters. The variable is returned with the APAR ID that caused this object to be patched. The variable is returned blank if the object has not been changed as a result of an APAR.

OBJAUD
Specifies the name of a 10-character variable used to return the auditing value of the object. The values that can be retrieved include *NONE, *USRPRF, *CHANGE, and *ALL. See the OBJAUD parameter on the Change Object Audit (CHGOBJAUD) command for more information.

Example

Jane Brown enters the following command to create a library:

```
CRTLIB  LIB(PGMLIB)  TYPE(*PROD)
  TEXT('Library for test programs')
```

Later, Jane, or anyone else with the proper authority, can retrieve the attributes of the library in a CL program as shown below:

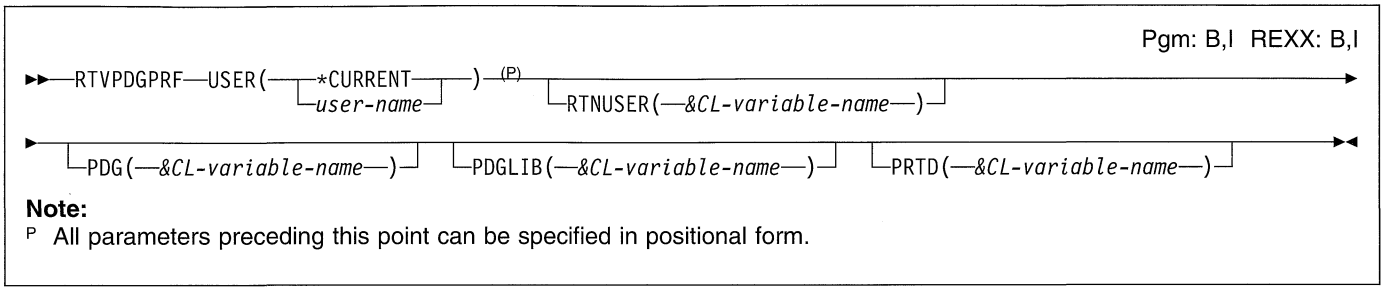
```
DCL  &DATE   CHAR  13
DCL  &OWN    CHAR  10
DCL  &STORE  CHAR  10
       .
       .
       .
RTVOBJD   OBJ(*LIBL/PGMLIB)  OBJTYPE(*LIB)  TEXT(&TEXT)
  CRTDATE(&DATE)  OWNER(&OWN)  STG(&STORE)
```

The values returned in the variables of the CL program are shown below:

```
&TEXT     =  Library for test programs
&DATE     =  0900211130000
&OWN      =  JBROWN
&STORE    =  *KEEP
```

The value returned in the variable &DATE indicates that PGMLIB was created on the eleventh day of February, 1990, at 1300 hours. The value returned in the variable &OWN indicates that the library was created by user profile JBROWN. The value returned in the variable &STORE indicates that the data in PGMLIB has not been freed, and the library has not been suspended.

RTVPDGPRF (Retrieve Print Descriptor Group Profile) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Print Descriptor Group Profile (RTVPDGPRF) command is used in a CL program to retrieve one or more of the print descriptor group profile values associated with a user profile. The values are returned in the specified CL variables for the desired user.

Restriction: The program must have *OBJMGT authority to use this command.

Required Parameters

USER
Specifies the name of the user whose print descriptor group (PDG) profile is being retrieved.

***CURRENT:** The user profile under which the current job is running is used.

user-name: Specify the name of the user whose PDG profile is being retrieved.

Optional Parameters

RTNUSER
Specifies a 10-character CL variable used to retrieve the name of user profile for which information is requested.

PDG
Specifies a 10-character CL variable used to retrieve the print descriptor group of the user profile for which information is requested.

PDGLIB
Specifies a 10-character CL variable used to retrieve the library of print descriptor group of the user profile for which information is requested.

PRTD
Specifies a 256 character CL variable used to retrieve the print descriptor name from the user profile for which information is requested.

Example

Assume a user with *OBJMGT authority entered the following command:

```
CHGUSRPRF  USER(JWONG)  PDG(*LIBL/TAXFORMS)  PRTD(FORM_C1)
```

Also assume the program with *OBJMGT authority contains the following commands and declarations:

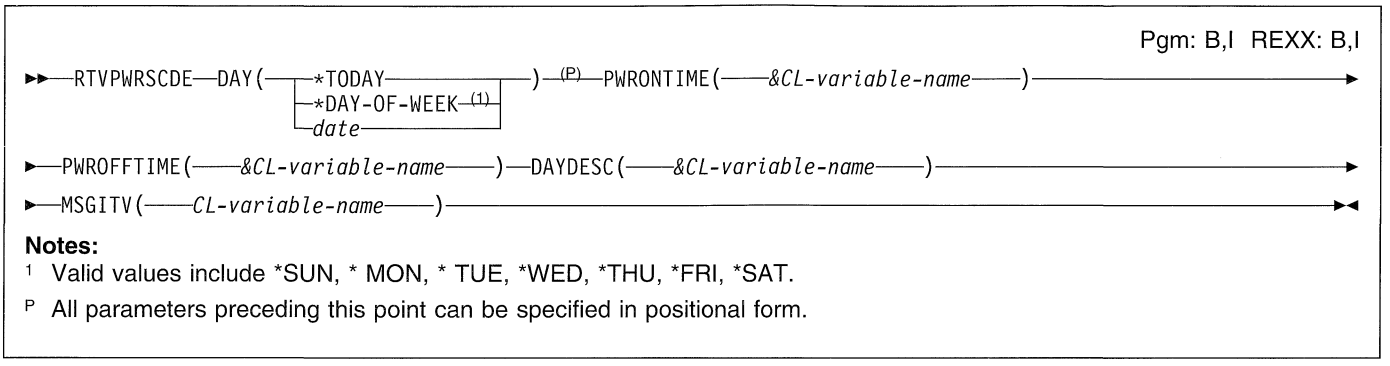
```
DCL  VAR(&USER)       TYPE(*CHAR)  LENGTH(10)
DCL  VAR(&GROUP)      TYPE(*CHAR)  LENGTH(10)
DCL  VAR(&LIBRARY)    TYPE(*CHAR)  LENGTH(10)
DCL  VAR(&DESCRIPT)   TYPE(*CHAR)  LENGTH(256)

RTVPDGPRF  USER(JWONG)  RTNUSER(&USER)  PDG(&GROUP)
  PDGLIB(&LIBRARY)  PRTD(&DESCIPT)
```

When the above program is called, the following values are returned:

```
&USER       'JWONG_ _ _ _ _'
&GROUP      'TAXFORMS_ _'
&LIBRARY    'TAXLIB_ _ _ _'
&DESCRIPT   'FORM_C1_..._'
```

Note: The underlines (_) above represent blanks. The value returned in variable &DESCRIPT is FORM_C1 followed by 249 blanks.

RTVPWRSCDE (Retrieve Power On/Off Schedule Entry) Command



Notes:
1 Valid values include *SUN, * MON, * TUE, *WED, *THU, *FRI, *SAT.
P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Power On/Off Schedule Entry (RTVPWRSCDE) command is used to retrieve a power on/off schedule entry value for use in a CL or REXX program. The value is returned (copied) to the specified CL variable in the program.

Optional Parameters

DAY
Specifies the day for which the user is retrieving a power on/off schedule entry.

***TODAY:** The current date's schedule entry is retrieved.

***DAY-OF-WEEK:** The default values for the specified day of the week are retrieved.

date: Specify the date for which a schedule entry is to be retrieved. The date must be specified in the format as defined by the job.

PWRONTIME
Specifies the name of the CL variable that receives the power on time. The variable named has a minimum length of 6 characters. The special value *NONE or the time in the format hhmmss, where hh = hours, mm = minutes, and ss = seconds, is returned.

PWROFFTIME
Specifies the name of the CL variable that receives the power off time. The variable named has a minimum length of 6 characters. The special value *NONE or the time in the format hhmmss, where hh = hours, mm = minutes, and ss = seconds, is returned. *NONE or the time in the format hhmm, where hh = hours and mm is minutes, is returned.

DAYDESC
Specifies the name of the CL variable that receives the specific date description. The variable named has a minimum length of 38 characters.

MSGITV
Specifies the name of the CL variable that receives the message interval value. The message interval is the number of minutes before the scheduled power off that a message is sent to all work stations warning users of the impending power off. The variable named has a minimum length of 2 characters.

Examples

Example 1: Retrieving Today's Schedule Entry

```
DCL VAR(&ONTIME) TYPE(*CHAR) LEN(6)
DCL VAR(&OFFTIME) TYPE(*CHAR) LEN(6)

RTVPWRSCDE   DAY(*TODAY)  PWRONTIME(&ONTIME)
  PWROFFTIME(&OFFTIME)
```

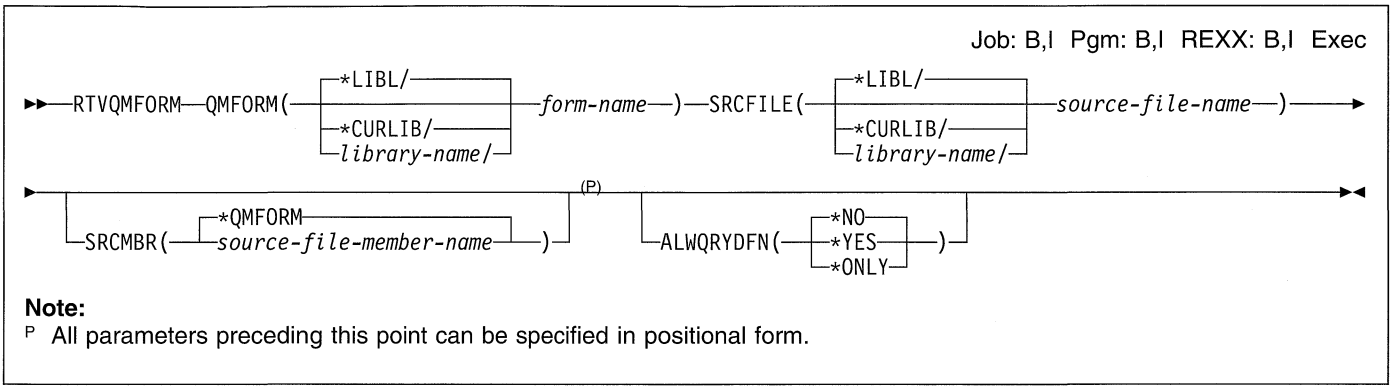
This command retrieves the power on and off times for today.

Example 2: Retrieving Tuesday's Schedule Entry

```
DCL VAR(&ONTIME) TYPE(*CHAR) LEN(6)
DCL VAR(&OFFTIME) TYPE(*CHAR) LEN(6)

RTVPWRSCDE   DAY(*TUE)  PWRONTIME(&ONTIME)
  PWROFFTIME(&OFFTIME)
```

This command retrieves the defaults for Tuesday's schedule.

RTVQMFORM (Retrieve Query Management Form) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Query Management Form (RTVQMFORM) command allows the user to retrieve encoded form source from a query management form (QMFORM) object. The source records are placed into a source file member that can be edited.

Form source can also be retrieved from a query definition (QRYDFN) object when the QMFORM specified by the user does not exist.

Required Parameters

QMFORM

Specifies the name of the query management form object whose source is being retrieved.

| The name of the form can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

form-name: Specify the name of the form being retrieved.

SRCFILE

Specifies the qualified name of the previously created source physical file into which the encoded form source records are being written.

| The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

source-file-name: Specify the name of the source file that is to receive the form source.

Optional Parameters

SRCMBR

Specifies the name of the source physical file member into which the encoded form source records are being written. If a source file member name is not specified, then the form name specified on the QMFORM parameter is used.

If the member existed before running this command, it is cleared before any source statements are written into it. If the member does not exist, it is created.

***QMFORM:** The member name is the same as the form name specified on the QMFORM parameter.

source-file-member-name: Specify the name of the member to receive the form source.

ALWQRYDFN

Specifies whether form information is taken from a QRYDFN object when a query management form (QMFORM) object cannot be found using the object name specified by the user. Any information that has to be derived in this way is discarded when the command has completed processing. No query management object is created.

***NO:** Information is not taken from a QRYDFN object.

***YES:** Information is taken from a QRYDFN object when the specified QMQRY object is not found.

***ONLY:** Information is taken only from a QRYDFN object. Query management objects are ignored.

Examples

Example 1: Retrieving Encoded Form Source

```
RTVQMFORM  QMFORM(RPTLIB/SALFORM)  SRCFILE(FORMS)
  SRCMBR(EMPFORM)
```

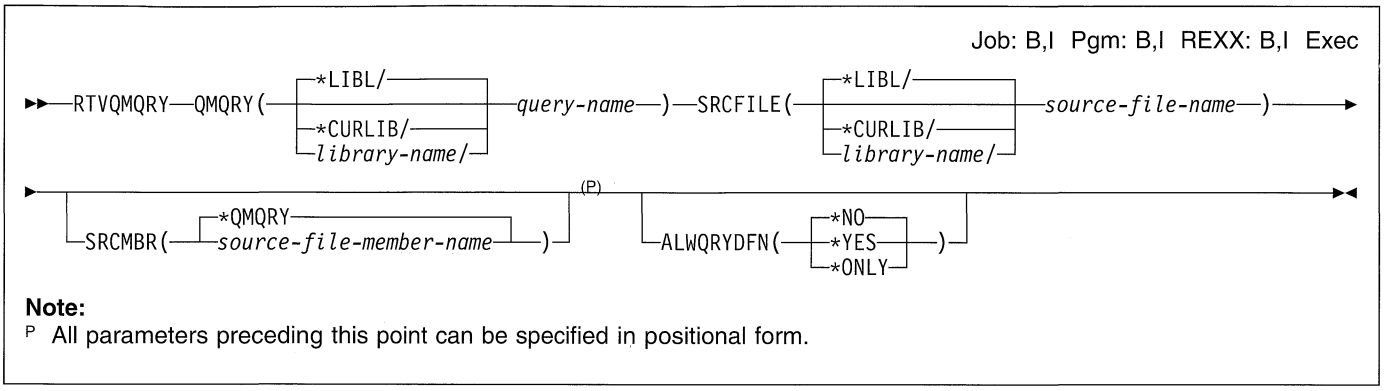
This command retrieves the encoded form source from the form named SALFORM located in the RPTLIB library. The encoded form source records that are retrieved are placed into the newly created or cleared member EMPFORM in the first file named FORMS in the user's library list.

Example 2: Retrieving Source From Either the QMFORM or the QRYDFN

```
RTVQMFORM  QMFORM(RPTLIB/SALFORM)  SRCFILE(FORMS)
  SRCMBR(EMPFORM) ALWQRYDFN(*YES)
```

This command retrieves the encoded form source from the query management form (QMFORM) named SALFORM located in the RPTLIB library. If there is no QMFORM object named SALFORM in the RPTLIB library, then the form source is retrieved from the query definition (QRYDFN) named SALFORM in the RPTLIB library. The encoded form source records that are retrieved are placed into the first file named FORMS in the user's library list.

RTVQMQRY (Retrieve Query Management Query) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Query Management Query (RTVQMQRY) command allows the user to retrieve Structured Query Language (SQL) source from a query management query (QMQRY) object. The source records are placed into a source file member that can be edited.

The user can also retrieve query source from a query definition (QRYDFN) object when the QMQRY specified by the user does not exist.

Required Parameters

QMQRY

Specifies the name of the query management query whose source is being retrieved.

| The name of the query can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

query-name: Specify the name of the query being retrieved.

SRCFILE

Specifies the qualified name of the previously created source physical file into which the query source records are being written.

| The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

source-file-name: Specify the name of the source file to receive the query source being retrieved.

Optional Parameters

SRCMBR

Specifies the name of the member into which the query source records are being written. If a source file member name is not specified, then the query name specified on the QMQRY parameter is used.

If the member existed before running this command, it is cleared before any source records are written to it. If the member does not exist, it is created.

***QMQRY:** The member name is the same as the query name specified on the QMQRY parameter.

source-file-member-name: Specify the name of the member to receive the query source.

ALWQRYDFN

Specifies whether query information is taken from a QRYDFN object when a QMQRY object cannot be found using the object name specified by the user. Any information that has to be derived in this way is discarded when the command has completed processing. No query management object is created.

***NO:** Information is not taken from a QRYDFN object.

***YES:** Information is taken from a QRYDFN object when the specified QMQRY object is not found.

***ONLY:** Information is taken only from a QRYDFN object. Query management objects are ignored.

Examples

Example 1: Retrieving SQL Source

```
RTVQMQRY  QMQRY(RPTLIB/SALQRY)  SRCFILE(QRYS)
  SRCMBR(EMPQRY)
```

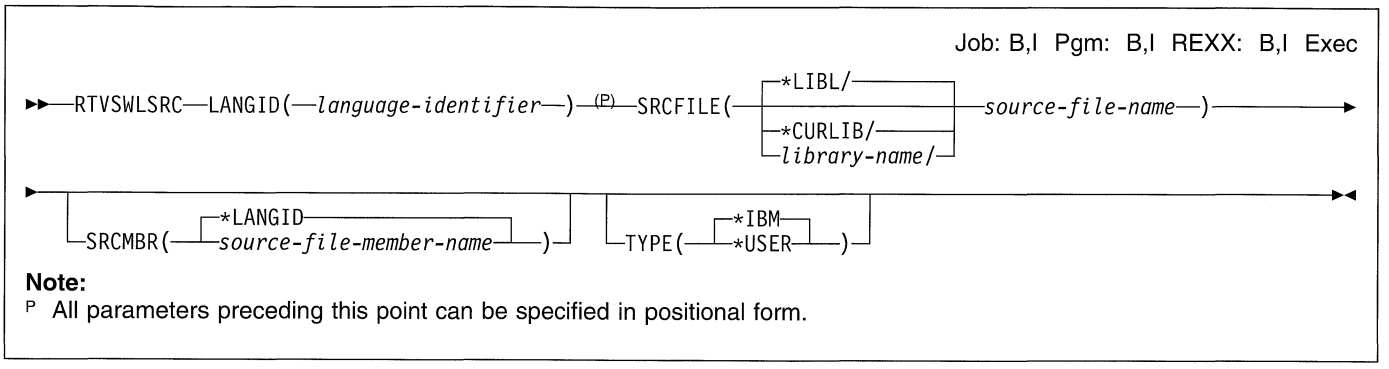
This command retrieves the source from the query named SALQRY located in the RPTLIB library. The source records that are retrieved are placed into the newly created or cleared member EMPQRY in the first file named QRYS in the user's library.

Example 2: Retrieving Source From Either the QMQRY or the QRYDFN

```
RTVQMQRY  QMQRY(RPTLIB/SALQRY)
  SRCFILE(QRYS)  SRCMBR(EMPQRY)
  ALWQRYDFN(*YES)
```

This command retrieves the source from the query management query (QMQRY) named SALQRY in the RPTLIB library. If there is no QMQRY object named SALQRY in the RPTLIB library, then the query source is retrieved from the query definition (QRYDFN) named SALQRY in the RPTLIB library. The source records are placed into the newly created or cleared member EMPQRY in the first file named QRYS in the user's library list.

RTVSWLSRC (Retrieve Stop Word List Source) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Stop Word List Source (RTVSWLSRC) command is used to retrieve the words from an IBM-supplied or user-created stop word list into a source file.

Required Parameters

LANGID

Specifies the language identifier (ID) for the stop word list.

SRCFILE

Specifies the qualified name of the source file used to receive the stop word list words. The contents of the source file are replaced.

The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

source-file-name: Specify the name of the source file.

Optional Parameters

SRCMBR

Specifies the name of the source file member used to receive the stop word list words. The member is in the source file specified on the SRCFILE parameter.

***LANGID:** The language ID is used as the source file member name.

source-file-member-name: Specify the name of the member in the source file used to receive the stop word list.

TYPE

Specifies the type of stop word list being retrieved

***IBM:** The stop word list is IBM-supplied.

***USER:** The stop word list is user-created.

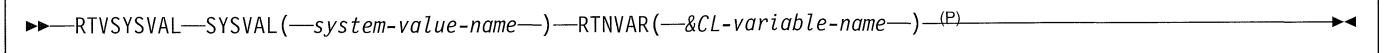
Example

```
RTVSWLSRC  LANGID(ENG)  SRCFILE(MYLIB/MYFILE)
```

This command retrieves the stop word list into source file MYFILE in library MYLIB that has the language ID ENG.

RTVSYSVAL (Retrieve System Value) Command

Pgm: B,I REXX: B,I



Note:
(P) All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve System Value (RTVSYSVAL) command is used in a CL program to retrieve the value from the specified system value so that it can be used in the program. The value is returned (copied) to the specified CL variable in the program.

Restrictions:

1. This command is valid only in compiled CL programs.
2. The attributes of the system value and the receiving CL variable must be compatible.
3. You must have *ALLOBJ and *SECADM special authorities to change security related system values.

Required Parameters

SYSVAL

Specifies the name of the system value whose value is retrieved and returned for use in the program. The names and descriptions of the system values that can be specified are in the *Work Management Guide*.

Double-Byte Character Set Considerations:

In addition to the system values specified in the *Work Management Guide*, you can retrieve the value of QIGC, which indicates whether you installed the double-byte character set (DBCS) version of the AS/400 system. If the system value for QIGC is 1, then the DBCS version of the AS/400 system is installed on your system. If the system value for QIGC is 0, the DBCS version of the AS/400 system is not installed on your system.

RTNVAR

Specifies the name of the CL program variable that receives the system value being returned. The type and length for the CL variable when it was declared must be compatible with that of the system value being received. The attributes of individual system values are described in the *Work Management Guide*.

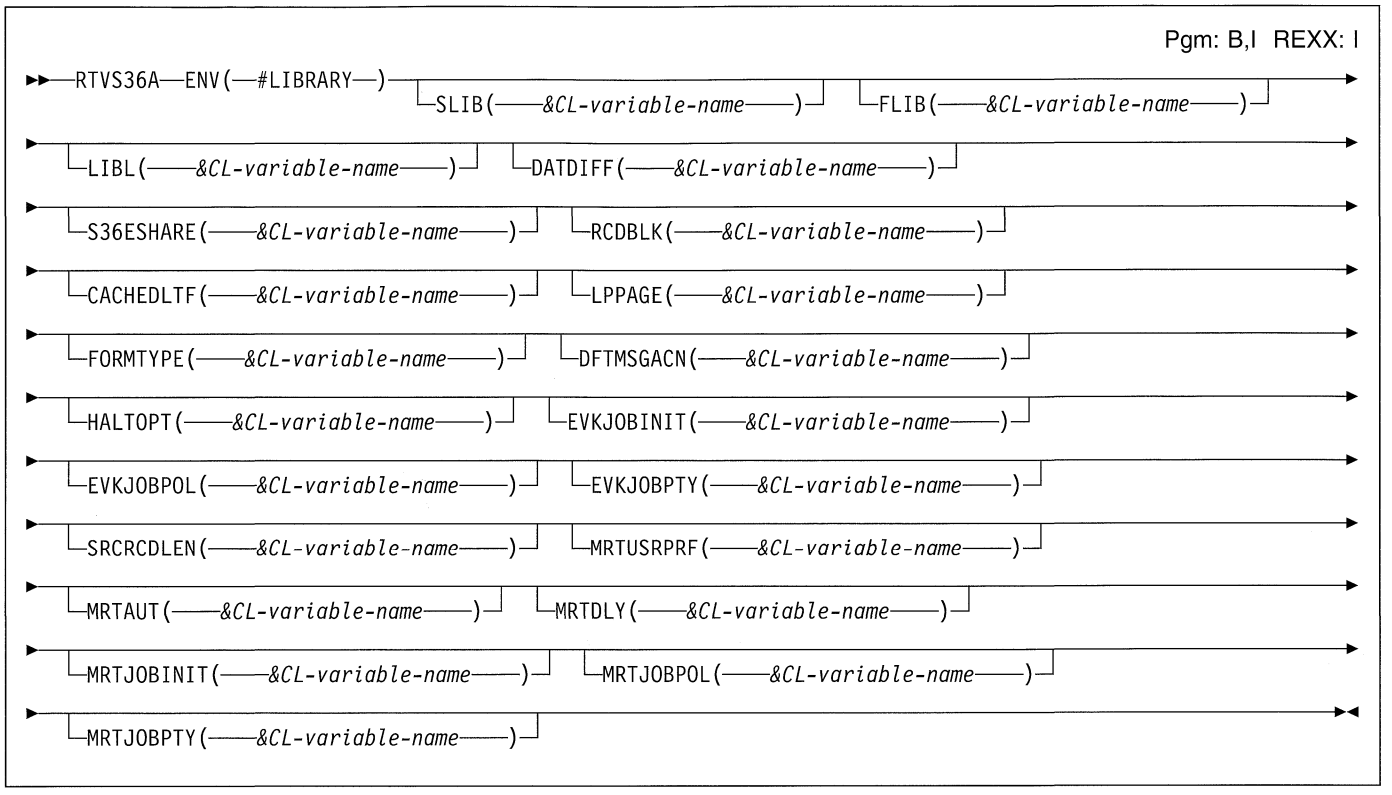
In general, the type of the return variable must match the type of the system value. For character system values that are 1 character long, the CL variable can be a character or logical variable. For character and logical system values, the length of the CL variable must equal the length of the system value. For decimal system values, the CL variable must have a length that is greater than or equal to the length of the system value.

Example

```
RTVSYSVAL  SYSVAL(QDATE)  RTNVAR(&DATE)
```

This command retrieves the date value from the system value QDATE and copies it into the CL variable &DATE. The CL variable must be declared as a 6-character variable to match the attributes of the QDATE system value.

RTVS36A (Retrieve System/36 Attributes) Command



Purpose

The Retrieve System/36 Attributes (RTVS36A) command allows the user to retrieve specific attribute information about the System/36 environment configuration and provide that information to a specified variable of a CL program or REXX procedure. Information about one or more attributes can be retrieved.

More information about the System/36 attributes that can be retrieved is in the description for the Change System/36 Attributes (CHGS36A) command.

Required Parameter

ENV
Specifies the name of the System/36 environment from which you are retrieving attributes. The value is #LIBRARY and cannot be changed.

Optional Parameters

SLIB
Specifies the name of an 8-character variable to receive the name of the default session library for jobs running in the System/36 environment.

FLIB
Specifies the name of a 10-character variable to receive the name of the default files library for jobs running in the System/36 environment.

LIBL
Specifies the name of a 4-character variable to receive information on whether the library list is used for jobs running in the System/36 environment. A value of *YES or *NO is returned in the variable.

DATDIFF
Specifies the name of a 4-character variable to receive information on whether files with the same name but different dates can be used for jobs running in the System/36 environment. A value of *YES or *NO is returned in the variable.

S36ESHARE
Specifies the name of a 4-character variable to receive information on whether programs share an open data path (ODP) to database files opened in the System/36 environment. A value of *YES or *NO is returned in the variable.

RCDBLK
Specifies the name of a 4-character variable to receive information on whether record blocking is used for sequential database files sharing an open data path in the System/36 environment. A value of *YES or *NO is returned in the variable.

CACHEDLTF
Specifies the name of a 4-character variable to receive information on whether deleted files are stored in a cache in the System/36 environment. A value of *YES or *NO is returned in the variable.

LPPAGE
Specifies the name of a 3-character variable to receive the number of lines printed on a page for jobs running in the System/36 environment. A value ranging from 1 through 112 is returned in the variable.

FORMTYPE
Specifies the name of a 4-character variable to receive the form type of the printer form used when printing a job in the System/36 environment. A value of *STD or a user-defined form type is returned in the variable.

DFTMSGACN
Specifies the name of a 9-character variable to receive the default message action used by the System/36 environment when an error occurs during the running of a CL command within a System/36 environment procedure. A value of *CONTINUE, *HALT, IGNORE, or *CANCEL is returned in the variable.

HALTOPT
Specifies the name of a 4-character variable to receive the options list for continuation after an error occurs in the System/36 environment and *HALT is specified for the default message action.

EVKJOBINIT
Specifies the name of a 6-character variable to receive the value for the method used to start System/36 EVOKE jobs or job steps in the System/36 environment. A value of *IMMED or *JOBQ is returned in the variable.

EVKJOBPOL
Specifies the name of an 8-character variable to receive the value for the storage pool used for jobs started with the *IMMED option in the System/36 environment. A value of *BASE or *CURRENT is returned in the variable.

EVKJOBPTY
Specifies the name of a 10-character variable to receive the value for the priority at which a job is started when it is started with the *IMMED option in the System/36 environment. A value ranging from 1 through 99 or the value *SUBMITTER is returned in the variable.

SRCRCDLEN
Specifies the name of a 3-character variable to receive the record length in bytes for System/36 source files QS36PRC and QS36SRC. A value ranging from 52 through 132 is returned in the variable.

MRTUSRPRF
Specifies the name of an 8-character variable to receive the user profile under which the Multiple Requester Terminal (MRT) program is running. A value of *OWNER or *FRSTUSR is returned in the variable.

MRTAUT
Specifies the name of an 8-character variable to receive the user authority to files used by the MRT program. A value of *ALLUSR or *FRSTUSR is returned in the variable.

MRTDLY
Specifies the name of a 5-character variable to receive the time in seconds that the system delays before ending the MRT program. A value ranging from 0 through 32767 is returned in the variable.

MRTJOBINIT
Specifies the name of a 6-character variable to receive the value for the method used to start an MRT job in the System/36 environment. A value of *IMMED or *JOBQ is returned in the variable.

MRTJOBPOL
Specifies the name of an 8-character variable to receive the value for the storage pool to be used for an MRT job started with the *IMMED option in the System/36 environment. A value of *BASE or *CURRENT is returned in the variable.

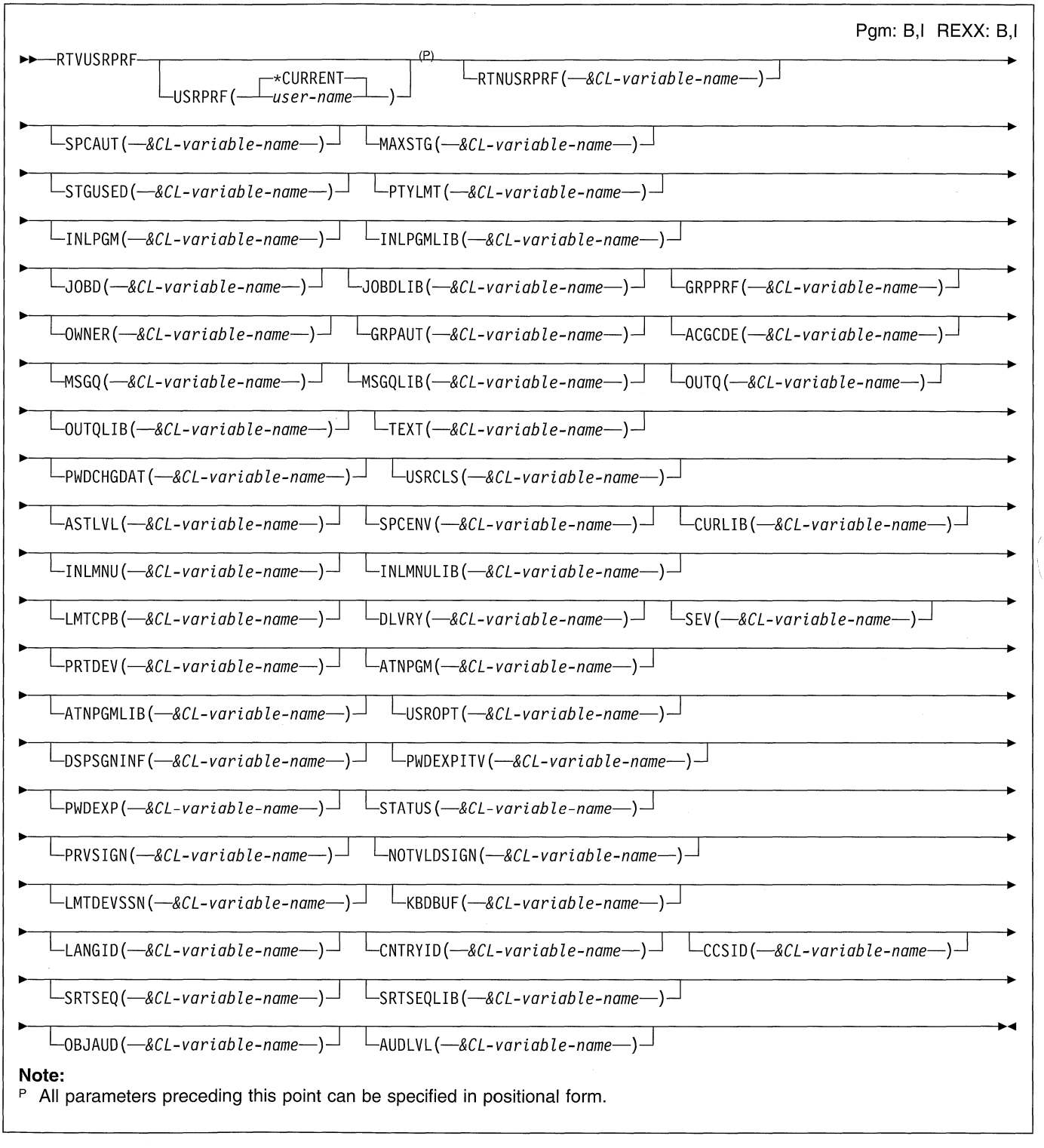
MRTJOBPTY
Specifies the name of a 10-character variable to receive the value for the priority to start an MRT job started with the *IMMED option. A value ranging from 1 through 99 or the value *SUBMITTER is returned in the variable.

Example

```
RTVS36A   RCBBLK(&RBLOCK)  HALT(&OPTION)  MRTUSRPRF(&USERI
```

This command retrieves the shared file record blocking value, the halt options list, and the user profile under which the MRT is running. The file record blocking value is copied into the CL variable &RBLOCK, which must be 4 characters in length. The halt options list is copied into the CL variable &OPTION, which must be at least 4 characters in length. The user profile under which the MRT is running is copied into the CL variable &USERID, which must be 8 characters in length.

RTVUSRPRF (Retrieve User Profile) Command



Note:

(P) All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve User Profile (RTVUSRPRF) command is used in a control language (CL) program or REXX procedure to retrieve one or more of the values stored and associated with a user. The values are returned in the specified variables for the desired user.

The CL prompt for this command lists the minimum length for the variables next to the appropriate parameters to be retrieved. A single digit is shown for character variables. Two digits are shown for decimal variables, where the first digit indicates the minimum variable length, and the second digit indicates the minimum number of decimal positions.

Restrictions: *READ authority is required for the user specified on the USRPRF parameter if any of the following parameters are specified: SPCAUT, MAXSTG, PTYLMT, GRPPRF, USRCLS, OWNER, GRPAUT, ACGCDE, TEXT, or PWDCHGDAT. If one of the above parameters is specified, and the job does not have *READ authority to the specified user, none of the values requested are returned.

Optional Parameters

USRPRF

Specifies the name of the user whose information is being retrieved. If a variable is specified, it must be 10 characters long and contain a user name or the value *CURRENT.

***CURRENT:** The user profile under which the current job is running is used.

user-name: Specify the name of the user whose information is being retrieved.

RTNUSRPRF

Specifies the name of a variable used to contain the name of the user profile whose information is retrieved. In CL programs, the variable has a length of 10 characters. If *CURRENT is specified on the USRPRF parameter, the value returned is the user profile name of the current job; if a name is specified, that name is returned for this parameter.

SPCAUT

Specifies the name of a variable used to retrieve the list of user special authorities. In CL programs, the variable has a length of 100 characters. The format returned is a list of up to 10 special authority entries, with each entry 10 characters long. If there are fewer than 10 special authorities in the list, the remaining entries are padded on the right with blanks. If the user has no special authorities, the first entry contains the value of *NONE followed by blanks.

The special authority list is returned in the following format:

```
Entry-1
   Special authority CHAR(10)
Entry-2
   Special authority CHAR(10)
     .
     .
     .
Entry-10
   Special authority CHAR(10)
```

MAXSTG

Specifies the name of a variable used to retrieve the maximum amount of auxiliary storage space that can be assigned to store permanent objects owned by the specified user. The value returned is either an 11-digit value or a value of –1 if it is *NOMAX. The variable must be an 11-digit value with zero decimal positions (11 0). For additional information on this parameter, see the Create User Profile (CRTUSRPRF) command.

STGUSED

Specifies the name of a variable that is used to retrieve the amount of auxiliary storage currently being used to store permanent objects owned by the specified user profile. In CL programs, the variable has a length of (15 0) characters. The value is returned in kilobytes (1 kilobyte (KB) equals 1024 bytes).

PTYLMT

Specifies the name of a variable used to retrieve the highest scheduling priority the user is allowed to have for each job submitted to the system. In CL programs, the variable has a length of 1 character. This value controls the job processing priority that any job running under this user can have. This means that values specified in the JOBPTY and OUTPTY parameters of any job command cannot exceed the PTYLMT value specified for the user under which the job is run. The scheduling priority can have a value ranging from 0 through 9, where 0 is the highest priority and 9 is the lowest priority.

INLPGM

Specifies the name of a variable used to retrieve the name of the initial program that starts when the specified user signs on the system. In CL programs, the variable has a length of 10 characters. If no initial program name is associated with the specified user, the value returned in the CL variable is *NONE. For more information on this parameter, see the Create User Profile (CRTUSRPRF) command.

INLPGMLIB

Specifies the name of a variable used to retrieve the name of the library that contains the initial program associated with the specified user. In CL programs, the variable has a length of 10 characters. If no initial program is associated with the specified user, blanks are returned in the variable.

JOBD
Specifies the name of a variable used to retrieve the name of the job description associated with the specified user. In CL programs, the variable has a length of 10 characters.

JOBDLIB
Specifies the name of a variable used to retrieve the name of the library that contains the job description associated with the specified user. In CL programs, the variable has a length of 10 characters.

GRPPRF
Specifies the name of a variable used to retrieve the name of the group profile. In CL programs, the variable has a length of 10 characters. If no group profile exists for the specified user profile, a value of *NONE is returned in the variable. For more information on this parameter, see the CRTUSRPRF (Create User Profile) command.

OWNER
Specifies the name of a variable used to retrieve the special value of *USRPRF or *GRPPRF. In CL programs, the variable has a length of 10 characters. This parameter indicates the owner of newly created objects. This is either the specified user or the user's group profile. If no group profile exists for the specified user profile, the value returned in the variable is *USRPRF.

GRPAUT
Specifies the name of a variable used to retrieve the type of authority granted to the group profile for newly created objects. In CL programs, the variable has a length of 10 characters. The special value of *NONE, *CHANGE, *ALL, *USE, or *EXCLUDE is returned in the variable. If there is no group profile for the specified user, the special value of *NONE is returned. If the group profile is the owner of the objects created by the specified user, the special value returned is *NONE. For more information on this parameter, see the CRTUSRPRF (Create User Profile) command.

ACGCDE
Specifies the name of a variable used to retrieve the value of the job accounting code assigned to the specified user. In CL programs, the variable has a length of 15 characters. If no job accounting code exists for the user profile, blanks are returned to the user. More information on this parameter is in the Create User Profile (CRTUSRPRF) command description.

MSGQ
Specifies the name of the CL variable that receives the group message queue name. This must be a character variable with a minimum length of 10 characters. If the message queue name has fewer characters than the variable allows, the value is padded on the right with blanks.

MSGQLIB
Specifies the name of a variable used to retrieve the name of the library containing the message queue associated with the specified user. In CL programs, the variable has a length of 10 characters.

OUTQ
Specifies the name of a variable used to retrieve the name of the output queue associated with the specified user. In CL programs, the variable has a length of 10 characters. The special value *DEV or *WRKSTN is returned in the variable.

OUTQLIB
Specifies the name of a variable used to retrieve the name of the library containing the output queue associated with the specified user. In CL programs, the variable has a length of 10 characters. Blanks are returned if the current value for OUTQ is *DEV or *WRKSTN.

TEXT
Specifies the name of a variable used to retrieve the text description of the user profile. In CL programs, this should be a 50-character variable. If there is no text associated with the user profile, blanks are returned in the CL variable.

PWDCHGDAT
Specifies the name of a variable used to retrieve the date of the last time the password for the specified user was changed. In CL programs, the variable has a length of 6 characters. The date is returned in the form YYMMDD. If the user does not have a password change date, blanks are returned.

USRCLS
Specifies the name of a variable used to retrieve the user class for the specified user. In CL programs, the variable has a length of 10 characters. The special value of *USER, *SYSOPR, *PGMR, *SECADM, or *SECOFR is returned in the variable.

ASTLVL
Specifies the assistance-level. The special value of *SYSVAL, *BASIC, *INTERMED, or *ADVANCED is returned as the variable.

SPCENV
Specifies the name of a variable for the starting environment for the specified user. In CL programs, the variable has a length of 10 characters. The special value of *SYSVAL, *NONE, or *S36 is returned in the variable.

CURLIB
Specifies the name of the variable used to retrieve the name of the job's default library for the specified user. In CL programs, the variable has a length of 10 characters. A value of *CRTDFT is returned in the variable if no current library exists for this user.

INLMNU
Specifies the name of a variable used to retrieve the name of the user's first menu when the specified user

signs on the system. In CL programs, the variable has a length of 10 characters. For more information on this parameter, see the CRTUSRPRF (Create User Profile) command.

INLMNULIB
Specifies the name of the variable used to retrieve the library name containing the initial menu. In CL programs, the variable has a length of 10 characters.

LMTCPB
Specifies the name of a variable used to retrieve the value for the limits to which users can change their user profiles and run commands. In CL programs, the variable has a length of 10 characters. The special value of *NO, *YES, or *PARTIAL is returned in the variable.

DLVRY
Specifies the name of a variable used to retrieve the message control delivery value for the specified user profile. In CL programs, the variable has a length of 10 characters. The special value of *NOTIFY, *BREAK, *HOLD, or *DFT is returned in the variable.

SEV
Specifies the name of a variable used to retrieve the message control severity level for the specified user. In CL programs, the variable has a length of (2 0) characters.

PRTDEV
Specifies the name of a variable used to retrieve the name of the printer device for the specified user. In CL programs, the variable has a length of 10 characters. A value of *SYSVAL is returned if the printer device name is from the system value QPRTDEV. A value of *WRKSTN is returned if the printer device name is from the printer device assigned to the user's work station.

ATNPGM
Specifies the name of a variable used to retrieve the name of the ATTN key handling program for the specified user. In CL programs, the variable has a length of 10 characters. A value of *SYSVAL or *NONE is returned.

ATNPGMLIB
Specifies the name of a variable used to retrieve the name of the library containing the ATTN key handling program for the specified user. In CL programs, the variable has a length of 10 characters. If *NONE is the current value of ATNPGM, blanks are returned in the variable.

USROPT
Specifies the name of a variable used to retrieve the list of user option values for the specified user. In CL programs, the variable has a length of 240 characters. The special value *NONE or a list of values is returned in the variable.

DSPSGNINF
Specifies the name of a variable used to retrieve the display sign-on information display indicator for the specified user. In CL programs, the variable has a length of 7 characters. The special value of *SYSVAL, *YES, or *NO is returned in the variable.

PWDEXPITV
Specifies the name of a variable used to retrieve the password expiration interval for the specified user. In CL programs, the variable specified must be packed (5 0) in length. The value returned is either a value ranging from 1 through 366, 0, if it is *SYSVAL, or –1 if it is *NOMAX.

PWDEXP
Specifies the name of a variable used to retrieve the password expired indicator for the specified user. In CL programs, the variable has a length of 4 characters. The special value of *YES or *NO is returned in the variable.

STATUS
Specifies the name of a variable used to retrieve the status of the specified user. In CL programs, the variable has a minimum length of 10 characters. The special value of *ENABLED or *DISABLED is returned.

PRVSIGN
Specifies the name of a variable used to retrieve the previous sign-on date and time for the specified user. In CL programs, the variable has a length of 13 characters. The date and time are returned in the form CYYMMDDHHMMSS. If the user does not have a previous sign-on date and time, blanks are returned.

NOTVLDSIGN
Specifies the name of a variable used to retrieve the number of sign-on attempts that were not valid for the specified user. In CL programs, the variable specified must be packed (11 0) in length.

LMTDEVSSN
Specifies the name of a variable used to retrieve the limit device sessions indicator for the specified user. In CL programs, the variable has a length of 7 characters. The special value of *SYSVAL, *YES, or *NO is returned in the variable.

KBDBUF
Specifies the name of a variable used to retrieve the keyboard buffering value for the specified user. In CL programs, the variable has a length of 10 characters. The special value of *SYSVAL, *NO, *TYPEAHEAD, or *YES is returned in the variable.

LANGID
Specifies the name of a variable that is used to retrieve the language identifier for the specified user. The value returned is either *SYSVAL or the 3-character language identifier. If *SYSVAL is returned, the language identifier for the user is determined by the QLANGID system

value. In CL programs, the variable has a length of 10 characters.

CNTRYID
Specifies the name of a variable used to retrieve the country identifier for the specified user. The value returned is either *SYSVAL or the 2-character country identifier. If *SYSVAL is returned, the country identifier for the user is determined by the QCNTRYID system value. In CL programs, the variable has a length of 10 characters.

CCSID
Specifies the name of a variable used to retrieve the coded character set identifier (CCSID) for the specified user. The value returned is either a 5-digit value or a value of -2 if it is *SYSVAL. If *SYSVAL is returned, the CCSID for the user is determined by the QCCSID system value. In CL programs, this should be a decimal variable length of (5 0).

SRTSEQ
Specifies the name of a variable used to retrieve the sort sequence table for the specified user. The value returned is one of the following: *HEX, *LANGIDUNQ, *LANGIDSHR, *SYSVAL, or the 10-character table identifier. If *SYSVAL is returned, the table identifier for the user is determined by the QSRTSEQ system value.

SRTSEQLIB
Specifies the name of a variable used to retrieve the sort sequence table library for the specified user. The value returned is the 10-character library identifier. The variable is set to blanks unless a sort sequence table name is specified.

OBJAUD
Specifies the name of a 10-character variable used to retrieve the object auditing value for the specified user. The special value of *NONE, *CHANGE, or *ALL, as specified on the Change User Audit (CHGUSRAUD) command, is returned in the variable. When less than 10 characters are returned, the variable is padded on the right with blanks.

AUDLVL
Specifies the name of a variable used to retrieve the object auditing level for the specified user. In CL programs, the variable has a length of 640 characters. The format returned is a list of a maximum of 64 object auditing level entries, with each entry 10 characters long. If there are fewer than 64 object auditing level entries in the list, the remaining entries are padded on the right with blanks. If the user has no object auditing levels, the first entry contains the value of *NONE followed by blanks. If the user has object auditing levels, one or more of the following special values, as specified in the Change User Audit (CHGUSRAUD) command, is returned in the variable: *CMD, *CREATE, *DELETE, *JOBDTA, *OBJMGT, *OFCSRV, *PGMADP, *SAVRST, *SECURITY, *SERVICE, *SPLFDTA, or *SYSMGT.

The object auditing level list is returned in the following format:

Entry-1
```
Object auditing level CHAR(10)
```
Entry-2
```
Object auditing level CHAR(10)
.
.
.
```
Entry-64
```
Object auditing level CHAR(10)
```

Example

```
CRTUSRPRF  USRPRF(SMITH)  SPCAUT(*SAVSYS *SECADM)
  MAXSTG(*NOMAX)  PTYLMT(4)  INLPGM(*NONE)
  MSGQ(QGPL/SMITHMQ)  OUTQ(QGPL/QSMITH)
  TEXT('John Smith User Profile')
```

The user specifies the above command.

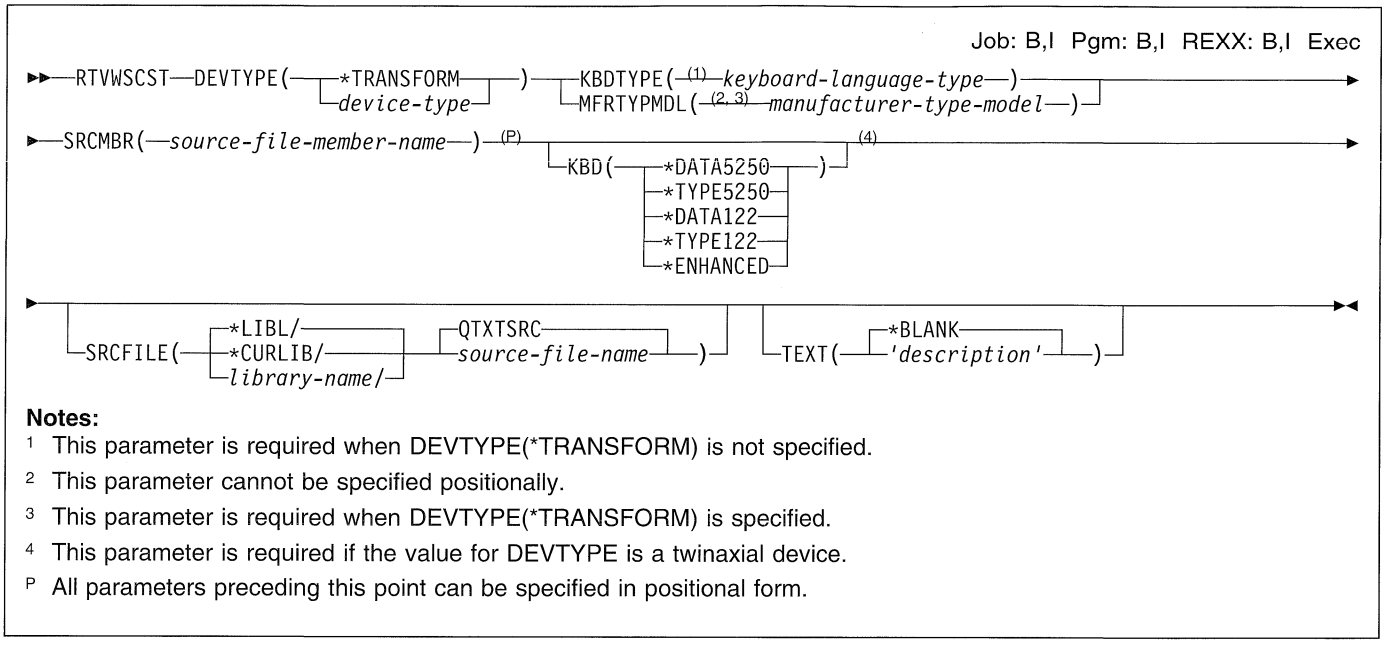
When user Smith calls a CL program containing the following:

```
DCL     &UNAME      *CHAR 10
DCL     &URIGHT     *CHAR 100
DCL     &IPGM       *CHAR 10
DCL     &IPGMLB     *CHAR 10
DCL     &UMSGQ      *CHAR 10
DCL     &UMSQLB     *CHAR 10
DCL     &USED       *CHAR 10

RTVUSRPRF  USRPRF(*CURRENT)  STGUSED(&USED)
  RTNUSRPRF(&UNAME)  SPCAUT(&URIGHT)  INLPGM(&IPGM)
  INLPGMLIB(&IPGMLB)
```

The following values are returned:

```
&UNAME      'SMITH      '
&URIGHT     |'*SAVSYS   *SECADM   (       ) ...    |
            |___________100 characters___________|
&IPGM       '*NONE      '
&IPGMLB     '           '
```

RTVWSCST (Retrieve Work Station Customizing Object Source) Command



Notes:

1 This parameter is required when DEVTYPE(*TRANSFORM) is not specified.

2 This parameter cannot be specified positionally.

3 This parameter is required when DEVTYPE(*TRANSFORM) is specified.

4 This parameter is required if the value for DEVTYPE is a twinaxial device.

P All parameters preceding this point can be specified in positional form.

Purpose

The Retrieve Work Station Customizing Object (RTVWSCST) command allows the user to retrieve a system-supplied set of table attributes for a given device type, keyboard language type, and keyboard type or a given manufacturer, type, and model of an ASCII printer into a source physical file member.

Required Parameters

DEVTYPE

Specifies the device type.

***TRANSFORM:** The SCS-to-ASCII host print transform function support is used by the ASCII printer.

device-type: Specify the device type to be used. See the *Workstation Customization Function Programmer's Guide* for a list of allowed device types.

MFRTYPMDL

Specifies the manufacturer, type, and model for an ASCII printer using host print transform function support. See Table 33 on the Create Device Description (Printer) (CRTDEVPRT) command for a list of the supported manufacturers, types, and models for ASCII printers using host print transform function support.

KBDTYPE

Specifies the 3-character country keyboard language identifier (used for EBCDIC and ASCII) for this display station.

See the KBDTYPE parameter on the Change Device Description (Display) (CHGDEVDSP) or the Create Device Description (Display) (CRTDEVDSP) for a list of the valid identifiers and the languages the identifiers represent. The ASCII device groups (if applicable) are also shown for each language.

SRCMBR

Specifies the name of the source file member to receive the retrieved table attributes.

Optional Parameters

KBD

Specifies the keyboard type.

***DATA5250:** A 5250 data entry keyboard is specified.

***TYPE5250:** A 5250 typewriter keyboard is specified.

***DATA122:** A 122 key data entry keyboard is specified.

***TYPE122:** A 122 key typewriter keyboard is specified.

***ENHANCED:** An enhanced keyboard is specified.

SRCFILE

Specifies the name of the source file in which a member is created to contain the retrieved table attributes. If the source file does not exist, it is created. The coded character set identifier for the source file is *HEX.

The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QTXTSRC: The IBM-supplied source file QTXTSRC is used.

source-file-name: Specify the name of the source file.

TEXT

Specifies text that briefly describes the source physical file member. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***BLANK:** Text is not specified.

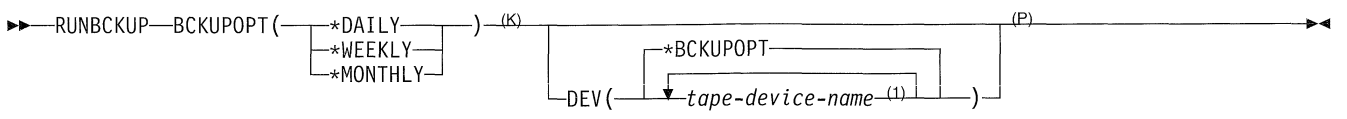
'description': Specify a description for the source physical file member.

Example

```
RTVWSCST  DEVTYPE(5251)  KBDTYPE(USB)  SRCMBR(MYSOURCE)
  KBD(*DATA5250)  SRCFILE(MYLIB/QTXTSRC)
```

This command retrieves the system mapping tables for a 5251 twinaxial display with a 5250 data entry type keyboard attached using the U.S. basic language. The tables are stored in source member MYSOURCE in source file QTXTSRC in library MYLIB.

RUNBCKUP (Run Backup) Command



Notes:

K All parameters preceding this point are key parameters.

1 A maximum of 4 repetitions

P All parameters preceding this point can be specified positionally.

Purpose

The Run Backup (RUNBCKUP) command allows the user to run a predefined backup of specified objects to tape. The backup may include libraries (all user libraries or those selected in the backup list), folders (all folders or root folders selected in the backup list), security data, configuration data, mail, and calendars.

Required Parameters

BCKUPOPT

Specifies the backup options to use.

***DAILY:** The daily backup options are used.

***WEEKLY:** The weekly backup options are used.

***MONTHLY:** The monthly backup options are used.

Optional Parameters

DEV

Specifies a list of tape devices to use for the backup.

***BCKUPOPT:** The tape device names stored in the specified options are used for the backup.

tape-device-name: Specify a list of tape devices used for the backup.

Examples

Example 1: Running a Daily Backup

```
RUNBCKUP  BCKUPOPT(*DAILY)
```

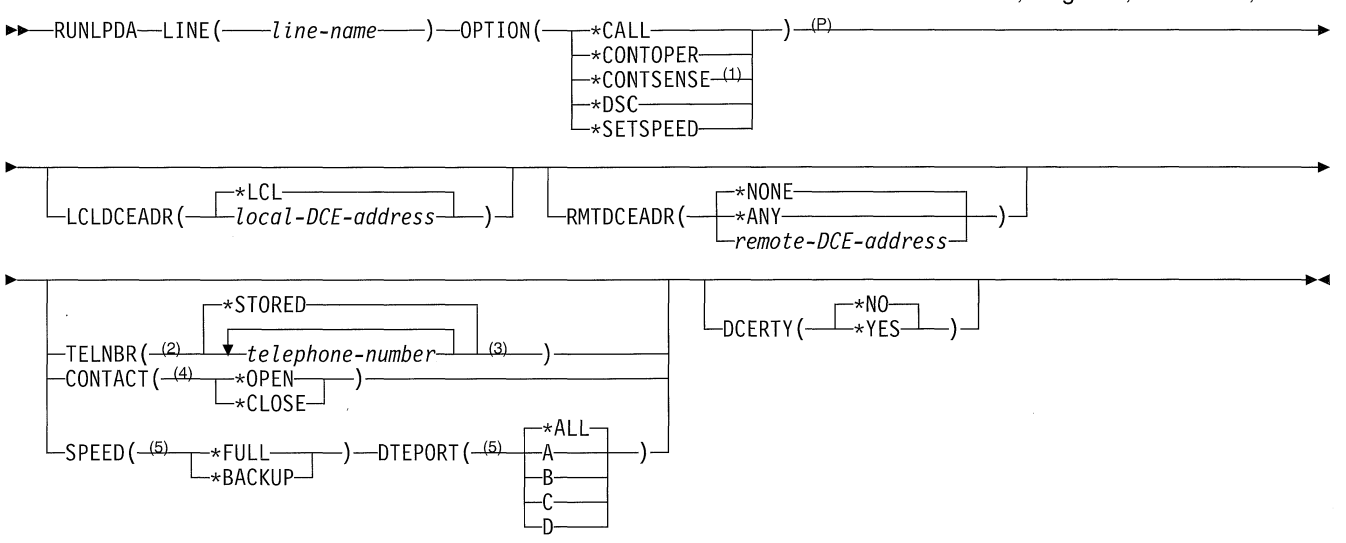
This command runs the daily backup using the devices specified in the options.

Example 2: Running a Monthly Backup

```
RUNBCKUP  BCKUPOPT(*MONTHLY)  DEV(TAP02)
```

This command runs the monthly backup using device TAP02 instead of those specified in the options.

RUNLPDA (Run LPDA-2) Command



Notes:

1 Not valid if RMTDCEADR(*NONE) is specified.

2 Valid only if OPTION(*CALL) is specified.

3 A maximum of 2 repetitions.

4 Valid only if OPTION(*CONTOPER) is specified.

5 Valid only if OPTION(*SETSPEED) is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Run LPDA-2 (RUNLPDA) command allows you to run a Link Problem Determination Aid-2 (LPDA-2) operational command on local or remote data circuit-terminating equipment (DCE). The RUNLPDA command can be used to:

- Establish or disconnect a switched telephone network connection.
- Open or close the relay contact in a coupler.
- Determine whether a relay contact is open or closed.
- Determine whether electric current is flowing through an internal sensor.
- Change the transmit speed of a DCE to full or backup.

The result of the RUNLPDA command is returned as a message.

Restrictions:

1. The RUNLPDA command is valid only for an analog LPDA-2 DCE attached to a nonswitched SDLC line.
2. This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Required Parameters

LINE

Specifies the name of the nonswitched SDLC line that is attached to the analog DCE on which the LPDA-2 operational command is to be run. If *CALL is specified on the OPTION parameter, the line must be varied on but not active. If *CONTOPER, *CONTSENSE, *DSC, or *SETSPEED is specified on the OPTION parameter, the line must be either varied on or active.

OPTION

Specifies which LPDA-2 operational command is run.

The contact operate, contact sense, and set transmit speed commands can be run on a local DCE or a remote DCE.

To run one of these commands on a local DCE:

- On the LCLDCEADR parameter, specify the address of the local DCE.
- On the RMTDCEADR parameter, specify *NONE.

To run one of these commands on a remote DCE:

- On the LCLDCEADR parameter, specify the address of the local DCE to which the remote DCE is connected.
- On the RMTDCEADR parameter, specify the address of the remote DCE.

Note: If the local DCE is configured as point-to-point secondary or multipoint tributary, LPDA-2 commands are not sent to the remote DCE.

***CALL:** The call out command is run. This command establishes a connection between a local and a remote DCE over a switched telephone network. This value is valid only if:

- Both the local DCE and the remote DCE have two-wire couplers installed or both the local DCE and the remote DCE have four-wire couplers installed.
- The line specified on the LINE parameter is varied on, but not active.

***CONTOPER:** The contact operate command is run. This command opens or closes the relay contact in the coupler, depending on the value specified on the CONTACT parameter. This value is valid only if a two-wire coupler is installed in the DCE on which this LPDA-2 command is run.

***CONTSENSE:** The contact sense command is run. This command reports whether the relay contact in the coupler is open or closed and whether electric current is flowing through the internal sensor. This option is valid only if a two-wire coupler is installed in the DCE on which this command is run.

***DSC:** The disconnect command is run. This command disconnects the switched telephone network connection between the local DCE and the remote DCE.

***SETSPEED:** The set transmit speed command is run. This command changes the transmit speed of the DCE to full or backup, depending on the value specified on the SPEED parameter. For multiport DCE configurations in which the data terminal equipment (DTE) ports can be set to different speeds, use the DTEPORT parameter to specify the port.

Note: The set transmit speed command may not change the transmit speed, depending on the configuration options selected for the DCE. Refer to the DCE documentation for more information.

Optional Parameters

LCLDCEADR

Specifies the hexadecimal address of the local DCE. Refer to the DCE documentation for more information on addressing.

***LCL:** X'01' is used for the address.

local-DCE-address: Specify the address of the local DCE. Valid values range from X'01' through X'FB'.

RMTDCEADR

Specifies the hexadecimal address of the remote DCE on which the LPDA-2 operational command is to be run.

***NONE:** X'00' is used as the address, which indicates that the LPDA-2 command is to be run on the local DCE.

***ANY:** X'FD' is used for the address. Specify this value in the following situations:

- The LPDA-2 command is to be run on any remote DCE connected to the local DCE.
- You do not know the remote DCE address on a point-to-point line.
- To run the LPDA-2 operational command on all tributary DCEs on a multipoint line. In this case, no detailed response is received.

remote-DCE-address: Specify the address of the remote DCE. Valid values range from X'01' through X'FB'.

TELNBR

Specifies the telephone number or telephone numbers that the local DCE dials to establish a connection to the remote DCE.

***STORED:** The telephone number or numbers stored in the local DCE when the DCE was configured are used.

telephone-number: Specify one telephone number if a two-wire coupler is installed in the local DCE. Specify two telephone numbers if a four-wire coupler is installed in the local DCE. Only numeric characters are processed by the LPDA-2 command, but you can also enter alphabetic characters or any other non-DBCS characters to improve readability. A comma (,) can be used to instruct the DCE to pause during dialing.

CONTACT

Specifies whether to open or close the relay contact in a two-wire coupler.

***OPEN:** The relay contact is opened.

***CLOSE:** The relay contact is closed.

SPEED

Specifies the desired transmit speed of the DCE.

***FULL:** The transmit speed is set to full.

***BACKUP:** The transmit speed is set to backup.

DTEPORT

Specifies the DTE port on the local or remote DCE for which the transmit speed is changed. This parameter is applicable only to multiport DCEs that do not use the multi-address configuration option.

***ALL:** The aggregate speed of the DCE is changed. Refer to the DCE documentation for information on the effect of changing the aggregate speed on the transmit speed of individual ports.

A: The transmit speed of the A-port is changed.

B: The transmit speed of the B-port is changed.

C: The transmit speed of the C-port is changed.

D: The transmit speed of the D-port is changed.

DCERTY

Specifies whether the local DCE resends the LPDA-2 command to the remote DCE if no response is received from the remote DCE. No retry can be attempted if OPTION(*CALL) or OPTION(*DSC) is specified.

***NO:** No retry is attempted.

***YES:** One retry is attempted.

Examples

Example 1: Establishing a Switched Telephone Network Connection

```
RUNLPDA   LINE(SDLCLINE)  OPTION(*CALL)
  LCLDCEADR(*LCL)  RMTDCEADR(*NONE)
  TELNBR(*STORED)
```

This command runs the call out command. The local DCE with address X'01' (*LCL) on line SDLCLINE dials the telephone numbers that are stored in the local DCE.

Example 2: Establishing a Switched Telephone Network Connection

```
RUNLPDA   LINE(SDLCLINE)  OPTION(*CALL)
  LCLDCEADR(*LCL)  RMTDCEADR(*NONE)
  TELNBR('9, 1-507-555-1212' '9, 1 (507) 555-1313')
```

This command runs the call out command. The local DCE dials the two numbers specified on the TELNBR parameter. The comma (,) indicates a pause during dialing. Other non-numeric characters are ignored, but are allowed for easier reading.

Example 3: Disconnecting a Switched Telephone Network Connection

```
RUNLPDA   LINE(SDLCLINE)  OPTION(*DSC)
  LCLDCEADR(10)  RMTDCEADR(*ANY)
```

This command runs the disconnect command. The local DCE with address X'10' disconnects from the switched telephone network.

Example 4: Closing the Relay Contact in the Local DCE

```
RUNLPDA   LINE(SDLCLINE)  OPTION(*CONTOPER)
  LCLDCEADR(02)  RMTDCEADR(*NONE)
  CONTACT(*CLOSE)
```

This command runs the contact operate command. The local DCE with address X'02' closes the relay contact in its two-wire coupler.

Example 5: Reporting the Status of the Relay Contact

```
RUNLPDA   LINE(SDLCLINE)  OPTION(*CONTSENSE)
  LCLDCEADR(01)  RMTDCEADR(04)
```

This command runs the contact sense command. A message reports the status of the relay contact in the remote DCE with address X'04'. (The correct local DCE address must be specified on the LCLDCEADR parameter.)

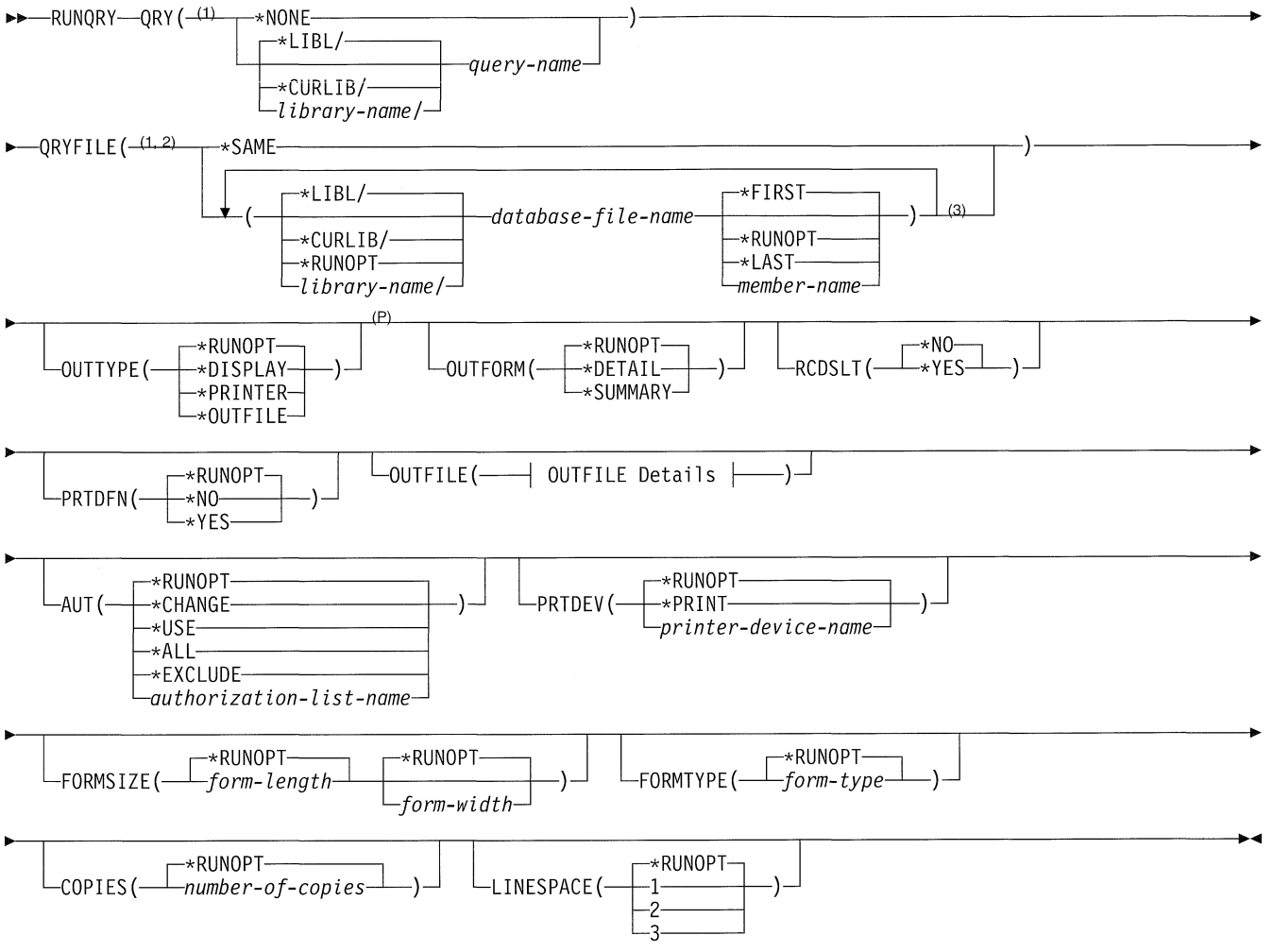
Example 6: Changing the Transmit Speed

```
RUNLPDA   LINE(SDLCLINE)  OPTION(*SETSPEED)
  LCLDCEADR(05)  RMTDCEADR(*NONE)
  SPEED(*BACKUP)  DTEPORT(B)
```

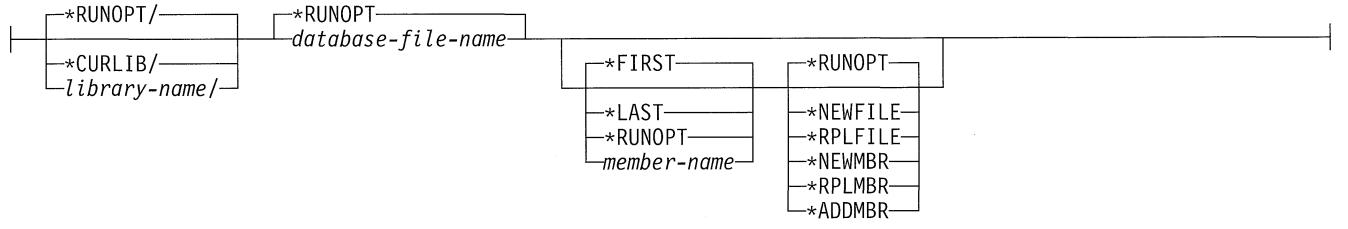
This command runs the set transmit speed command. The transmit speed for Port B of the local DCE with address X'05' is changed to backup speed.

RUNQRY (Run Query) Command

Job: B,I Pgm: B,I REXX: B,I Exec



OUTFILE Details:



Notes:

1 Either QRY or QRYFILE must be specified; or, both can be specified.

2 *RUNOPT is the default value for the remaining parameters. The parameter will default to whatever was specified in the query definition if a query name is entered. If only a file name is entered, the default will be the first in the list or first after *RUNOPT when *RUNOPT is first.

3 A maximum of 32 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Run Query (RUNQRY) command runs an existing query or a default query if only a file name is specified for this command. The query gets information from the system database and produces a report of that information. The report is created in either detailed or summary form. The definition of the query can be printed when output to a printer or database file is specified. The output is shown, printed, or stored in a database file.

The command is used in three ways: to run an existing query (one that has already been created), to run an existing query with some of its values changed by values specified on this command, or to run a default query based only on the defaults and values specified in this command.

- To run an existing query without changing the file or files to query, use the QRY parameter (without the QRYFILE parameter) to specify the name of the query.
- To run a changed version of an existing query, use the QRY parameter and the appropriate parameters to change the definition as desired. The parameter values specified on this command override the corresponding values in the existing query definition, but only when the command is processing. For example, use the QRYFILE parameter to indicate a different file or list of files to use in the query.
- To query a file and without a previously defined query definition, use the QRYFILE parameter to specify which file to query. Only one file name can be specified for a default query.

If the user specifies both the QRY and QRYFILE parameters, the files specified in the QRYFILE parameter override the file names specified in the query. Therefore, if multiple files (and members) are defined in the query definition and the user wants to change one or two of them, specify *SAME for the file selections that do not change, and specify the values for the files to override.

Notes:

1. When a changed version of an existing query is run, the changes specified on the RUNQRY command do *not* change any of the values in the query definition itself; they affect only the results of the report being run.
2. If an existing query (identified on the QRY parameter) is used, *RUNOPT is the default value for any unspecified parameters; that is, the same value specified (or assumed) in the definition of the query is used as the default. If this is a default query, the default value is not *RUNOPT, but is the next predefined value listed in the syntax diagram.

Required Parameters

QRY

Specifies the name of an existing query to run. If QRY is not specified, QRYFILE must be specified.

***NONE:** No existing query definition is used. Instead, a default query (or quick query) is used to get information from the file specified on the QRYFILE parameter.

If no value is specified, the file or files that were specified when the query was defined are the files to be queried.

The name of the query can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

query-name: Specify the name of the query to run.

QRYFILE

Specifies the database file or files to be queried for information. If the QRY parameter is specified, as many as 32 files can be specified on this parameter by using the file names and/or using the default value *SAME for one or more of the files. If the QRY parameter is not specified, only one file name can be specified on this parameter. If QRYFILE is not specified, QRY must be specified.

If no value is specified, the file or files that were specified when the query was defined are the files used to run the query.

***SAME:** The value does not change.

Element 1: Database File Name

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***RUNOPT:** The library specified in the query definition for the file selection is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the names of one or more database files that contain the data from which the system gets information to produce the output. Up to 32 files can be specified.

Element 2: Database File Member

***FIRST:** The first member in the file is the member to query.

***RUNOPT:** The member specified in the query definition for this file selection is used.

***LAST:** The last member in the file is the member to query.

member-name: Specify the name of the file member to query.

Optional Parameters

OUTTYPE

Specifies where the report or output produced by the query is sent. If no value was specified in the query name on the command, or if a query name is not specified, *DISPLAY is assumed.

***RUNOPT:** If a query definition is being used, the type of output specified in the query definition is the type produced when this query is run.

***DISPLAY:** The output produced by the query is sent to the display station that runs the command. If the command is being run in batch, the output is sent to the printer and not to the display.

***PRINTER:** The output produced by the query is printed.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

OUTFORM

Specifies the form of output produced by the query. If no value was specified in the query, and none was entered on the command, or if a query name is not specified, *DETAIL is assumed.

***RUNOPT:** If a query definition is used, the output form specified in the query definition is used when this query is run.

***DETAIL:** The output form produced by the query is a report containing detail records and summary records if any exist.

***SUMMARY:** The output form produced by the query is a report containing summary records only.

RCDSLT

Specifies whether the query is run with a run time selection test.

Note: The AS/400 Query/400 licensed program must be installed to specify *YES.

***NO:** The query is run without showing the record selection display in Query.

***YES:** Record selection definition is allowed for this run only. A display is shown on which the user can change the record selection tests defined in the query or specify record selection tests if a query name was not specified. If *YES is specified, the query will run interactively.

PRTDFN

Specifies whether the query definition is printed with the report when the query is run. The definition can be printed when the output of the query is printed or stored in a data base file, as determined by the OUTTYPE parameter. If a value is not specified in the query or in this parameter, or if a query name is not specified, the parameter *NO is assumed.

***RUNOPT:** If a query definition is used when the query is run, the print option specified in the query definition is used.

***NO:** The query definition is not printed when the query is run.

***YES:** The query definition is printed in the report. *YES cannot be specified if OUTTYPE(*DISPLAY) is specified or assumed.

OUTFILE

Specifies the database file (if any) that receives the query output. If no value is specified for this parameter, the library, file, and member specified in the query are assumed. If a query is not specified, the file QQRYOUT is created in the current library (*CURLIB). The first member (*FIRST) of this new file is used for the output.

Note: If the user did not specify a current library, the QGPL library is used.

If the name specified by the OUTFILE parameter does not exist, the system creates it in the specified library.

Element 1: Database File Name

The name of the database file can be qualified by one of the following library values:

***RUNOPT:** If specified in the query definition, the output is directed to the database file named in the query definition.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

***RUNOPT:** The database file specified in the query is used the one that receives the output of the command.

database-file-name: Specify the name of the database file that receives the output of the command.

Element 2: Database File Member

***FIRST:** The first member in the file receives the query output.

***LAST:** The last member in the file receives the query output.

***RUNOPT:** The member specified in the query receives the query output.

member-name: Specify the name of the file member used to receive the query output.

Element 3: Adding or Replacing Data

The last option specifies whether to put the data in a new database file, replace an existing database file, add a new member, replace an existing member, or add data to an existing member. If no value is specified in the

query or in this parameter, or if a query name is not specified, the value *NEWFILE is assumed.

***RUNOPT:** If a query definition is used, the member option specified in the query definition is the type used when this query is run.

***NEWFILE:** The output is written to a new database file.

***RPLFILE:** The output deletes the old file and creates a new file.

***NEWMBR:** The output is added as a new member.

***RPLMBR:** The existing member is cleared and the output is then added.

***ADDMBR:** The output is added to the end of an existing member.

AUT

Specifies the authority given to users who do not have specific authority to the run query, who are not on an authorization list, and whose user group has no specific authority to the run query. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***RUNOPT:** If specified in the query definition, the authority named in the query definition is used.

***CHANGE:** The user can perform all operations on the object except those limited to the owner or controlled by object existence authority and object management authority. The user can change and perform basic functions on the object. Change authority provides object operational authority and all data authority.

***USE:** The user can perform basic operations on the run query, such as running a program or reading a file. The user cannot change the run query. *USE authority provides object operational authority and read authority.

***ALL:** The user performs all operations on the file except those limited to the owner or controlled by authorization list management authority.

***EXCLUDE:** The user cannot access the run query.

authorization-list-name: Specify the name of the authorization list used.

PRTDEV

Specifies the printer device on which the report is printed. If no value is specified, the printer that was specified when the query was defined is assumed. If no printer is specified in the query or in this parameter, or if a query name is not specified, the value *PRINT is assumed.

Note: If an override is in effect for the printer file QPQUPRFIL, this parameter uses the value specified by the override.

***RUNOPT:** If a query definition is used, the printer specified in the query definition is used to print the output when this query is run.

***PRINT:** The default printer, as defined by QPQUPRFIL, is used to print the output when this query is run.

printer-device-name: Specify the name of the printer that is used to print the output when this query is run.

FORMSIZE

Specifies the length and width of the forms on which the report is printed. If no value is specified in the query or in this parameter, or if a query name is not specified, 132 is the assumed form width, and the value from the file QPQUPRFIL is the assumed form length.

Note: If an override is in effect for the printer file QPQUPRFIL, this parameter uses the value specified by the override.

Element 1: Form Length

***RUNOPT:** If a query definition is being used when the query is run, the form size specified in the query definition is used. If the formsize specified in the query definition is blank, the value from QPQUPRFIL is assumed.

form-length: Specify the form length used when this query is run. Valid values range from 1 through 255.

Element 2: Form Width

***RUNOPT:** If a query definition is being used when the query is run, the form size specified in the query definition is used. If the formsize specified in the query definition is blank, the value from QPQUPRFIL is assumed.

form-width: Specify the form width used when this query is run. Valid values range from 1 through 198.

FORMTYPE

Specifies the type of form on which the output is printed. The identifiers used to indicate the type of forms are user-defined and can be a maximum of 10 characters in length.

Note: If a value is not specified in the query or on this parameter, or if a query name is not specified, the value in QPQUPRFIL is assumed. If an override is in effect for the printer file QPQUPRFIL, this parameter uses the value specified by the override.

***RUNOPT:** If a query definition is used, the form type specified in the query definition is used when this query is run.

form-type: Specify the form type that is used when this query is run.

COPIES

Specifies the number of copies being printed.

Note: If a value is not specified in the query or on this parameter, or if a query name is not specified, 1 is the assumed number of copies. If an override is in effect for the printer file QPQUPRFIL, this parameter uses the value specified by the override.

***RUNOPT:** If a query definition is used, the number of copies specified in the query definition is used when this query is run. If the number of copies specified in the query definition is blank, the number of copies from QPQUPRFIL is assumed.

number-of-copies: Specify the number of copies to print when this query is run. Specify a number ranging from 1 through 255.

LINESPACE

Specifies the number of blank lines to leave between lines in the report. The numbers range from 1 to 3. If a value was not specified in the query or on this parameter, or if a query name is not specified, 1 is the assumed value.

***RUNOPT:** If a query definition is used, the number of lines specified in the query definition is used when this query is run.

1: Indicates that single spacing (no blank lines) is used when the query output is printed.

2: Indicates that double spacing (1 blank line) is used when the query output is printed.

3: Indicates that triple spacing (2 blank lines) is used when the query output is printed.

Examples

Example 1: Printing Summary Records Only

```
RUNQRY  QRY(LIBX/QRY1)  OUTTYPE(*PRINTER)
  OUTFORM(*SUMMARY)  COPIES(4)
```

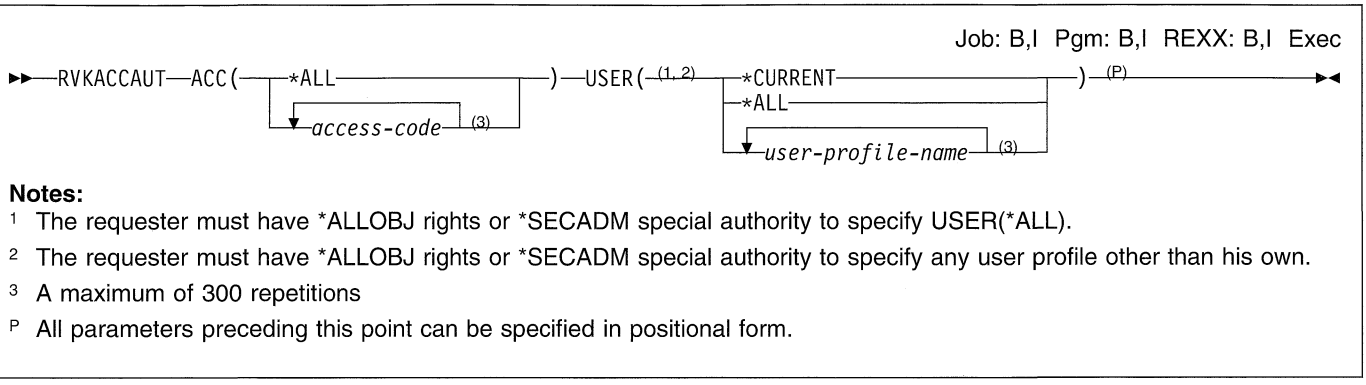
This command runs the query QRY1 located in library LIBX. The report that is produced and printed contains summary records only. Four copies of the report are printed.

Example 2: Running a Default Query

```
RUNQRY  QRYFILE((LIBX/FILE2 *FIRST))  OUTTYPE(*OUTFILE)
  OUTFORM(*DETAIL) RCDSLT(*YES)
  OUTFILE(LIB2/OUT1 MBR4)
  MBROPT(*NEWMBR)
```

This command runs a default query and gets the data from the first member of file FILE2 located in library LIBX. Member MBR4 is created as a new member to file OUT1 in library LIB2, and contains the output from the default query. The record selection display is shown to allow the user to specify which records from file FILE2 in library LIBX are written to new member MBR4 in file OUT1 in library LIB2. The output contains detail records only.

RVKACCAUT (Revoke Access Code Authority) Command



Notes:

1 The requester must have *ALLOBJ rights or *SECADM special authority to specify USER(*ALL).

2 The requester must have *ALLOBJ rights or *SECADM special authority to specify any user profile other than his own.

3 A maximum of 300 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Revoke Access Code Authority (RVKACCAUT) command takes away the access code authority for an individual user or a group of users.

Note: You can specify USER(*CURRENT) to remove access code authority from any access code to which you have authority.

Restriction: You must have *ALLOBJ authority to take away access code authority for other users.

Required Parameters

ACC

Specifies the access code for which authority is revoked. The access code is a decimal number ranging from 1 through 2047.

***ALL:** All access code authority for the user is revoked.

access-code: Specify a decimal number, ranging from 1 through 2047, that specifies the access code authority that is revoked.

USER

Specifies the name of the user profile for whom access code authority is revoked.

***CURRENT:** The user profile under which the current job is running is used.

***ALL:** Access code authority is revoked for all users.

user-profile-name: Specify the profile name of the user whose access code authority is revoked.

Examples

Example 1: Revoking Authority of Current User

```
RVKACCAUT   ACC(250)  USER(*CURRENT)
```

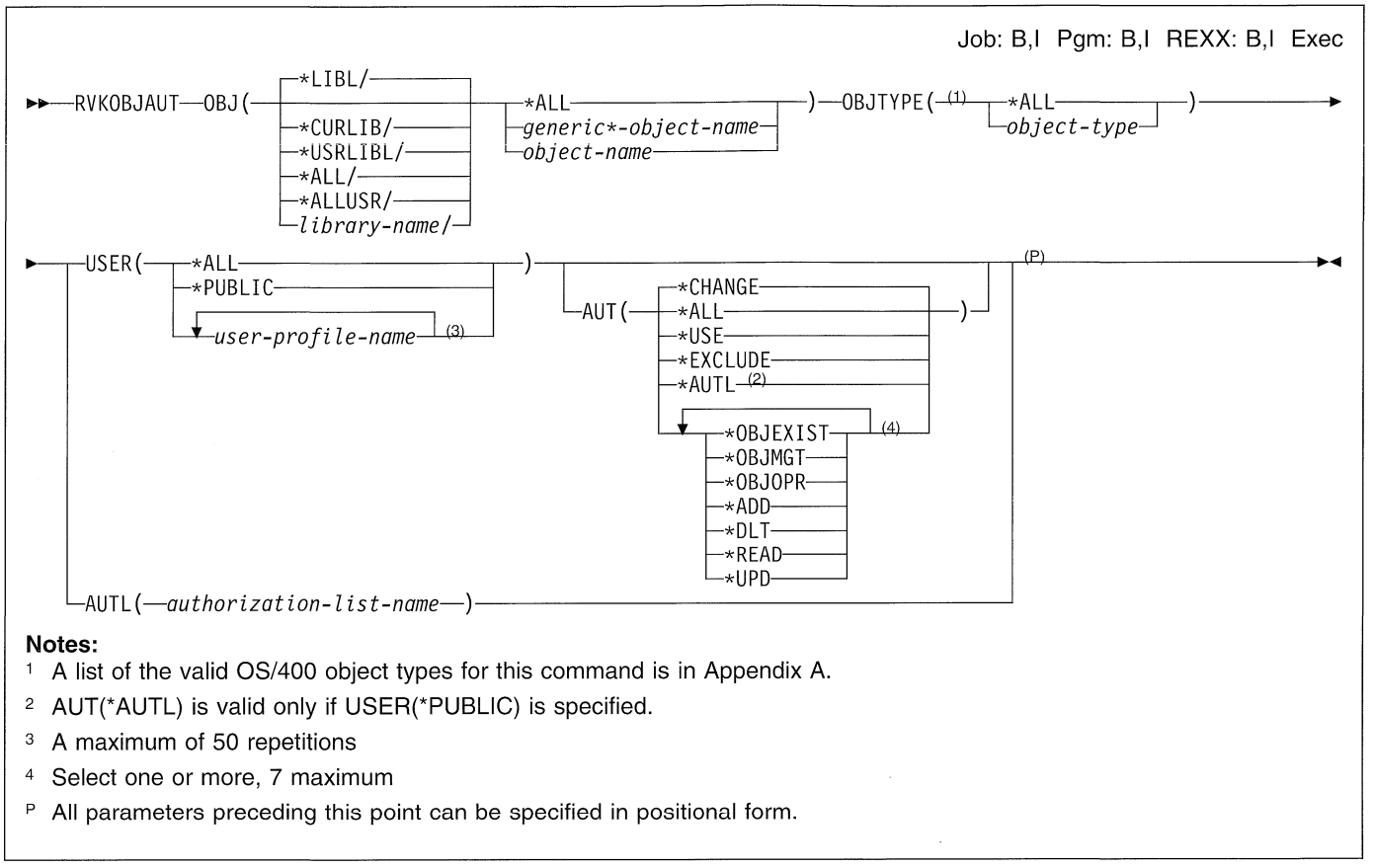
This command takes away the access code authority of access code 250 from the user currently running this command.

Example 2: Revoking Authority of Specific User

```
RVKACCAUT   ACC(300)  USER(BILLY)
```

This command takes away the access code authority of access code 300 from user BILLY. This command was run by someone with *ALLOBJ or *SECADM special authority, or by BILLY. A user who runs this command for himself can enter USER(*CURRENT) or his own user profile name; they are the same.

RVKOBJAUT (Revoke Object Authority) Command



Notes:

1 A list of the valid OS/400 object types for this command is in Appendix A.

2 AUT(*AUTL) is valid only if USER(*PUBLIC) is specified.

3 A maximum of 50 repetitions

4 Select one or more, 7 maximum

P All parameters preceding this point can be specified in positional form.

Purpose

The Revoke Object Authority (RVKOBJAUT) command is used to take away specific (or all) authority for the named objects from one or more users also named in the command, or to remove the authority of an authorization list for the named objects. This command can be entered by the security officer, by an object's owner, or by a user who has object management authority for the object being removed. A user with object management authority can remove only the authorities that that user has. A user may not be able to give or remove authorities for an object that has been allocated (locked) to another job. If a specific (not *ALL) authority cannot be revoked, a message is issued that indicates the authorities that were not revoked.

Restrictions:

1. Before this command is used to remove authorities to use a device, control unit, or line description, its associated device, control unit, or line must be varied on.
2. Authority to use a device cannot be revoked if a user is currently signed on to the device.

   Note: Users can revoke their own authority to a device if they are currently signed onto that device. However, doing so may produce unpredictable results and is not advisable.
3. For display stations or for work station message queues associated with the display station, if this command is not entered at the device for which authorities are being revoked, it should be preceded by the Allocate Object (ALCOBJ) command and followed by the Deallocate Object (DLCOBJ) command.
4. Object type *DOC cannot be specified. Document interchange support must be used.
5. Object type *AUTL cannot be specified. The Change Authorization List Entry (CHGAUTLE) or Remove Authorization List Entry (RMVAUTLE) commands must be used.
6. AUT (*AUTL) can be specified only with USER (*PUBLIC).
7. Only a user with *ALL authority or the owner can remove the authorization list.

Security Risk

Revoking all authorities specifically given to a user for an object can result in the user having more authority than before the revoke operation. If a user has *USE authority for an object and *CHANGE authority on the authorization list that secures the object, revoking *USE authority results in the user having *CHANGE authority to the object.

Required Parameters

OBJ

Specifies the qualified name of the objects for which specific authority is revoked. Either a specific or a generic object name is specified with a library name. If *ALL is specified, the name of a library must be specified. For more information on the use of generic functions, refer to "Rules for Specifying Names."

The name of the object can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries on the system are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All objects of the specified type (objtype) found in the search have specific authorities revoked. Specify the name of a library with *ALL.

generic-object-name:* Specify the generic name of the object. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be revoked only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

object-name: Specify the name of the object for which specific authorities are revoked.

OBJTYPE

Specifies the object type of the object that has specific authorities revoked. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ALL:** All object types have specific authorities revoked.

object-type: Specify the type of the object for which specific authorities are revoked.

USER

Specifies the names of one or more users whose specific authorities to the named object are being removed. If a user was given the authority by USER(*PUBLIC) being specified in the Grant Object Authority (GRTOBJAUT) command, the same authorities are revoked by *PUBLIC being specified in this parameter. Users who were given specific authority by having their names specified in the GRTOBJAUT command must have their names specified on this parameter to remove the same authorities.

Either this parameter or the AUTL parameter must be specified.

***ALL:** The authorities specified in the AUT parameter are taken away from all enrolled users of the system except the owner, if they are publicly or explicitly authorized.

***PUBLIC:** The specified authorities are taken away from users who do not have specific authority for the object, who are not on the authorization list, and whose group has no authority. Users who have specific authority still retain their authorities to the object.

user-profile-name: Specify the user names of one or more users that are having the specified authorities revoked. The authorities specified in the AUT parameter are being specifically taken away from each specified user. This parameter cannot be used to remove public authority from specific users; only authorities that were specifically given to them can be specifically revoked.

AUT

Specifies the authority given to users who do not have specific authority to the object, who are not on an authorization list, and whose user group has no specific authority to the object.

***CHANGE:** The user can perform all operations on the object except those limited to the owner or controlled by object existence authority and object management authority. The user can change and perform basic functions on the object. Change authority provides object operational authority and all data authority.

***ALL:** The user can perform all operations except those limited to the owner or controlled by authorization list management authority. The user can control the object's existence and specify the security for the object, change the object, and perform basic functions on the object. The user can change ownership of the object. If the

object is an authorization list, the user cannot add, change, or remove user IDs. Revoking *ALL authority from *PUBLIC causes the *PUBLIC authority to change to *EXCLUDE.

***USE:** The user can perform basic operations on the object, such as running a program or reading a file. The user cannot change the object. *USE authority provides object operational authority and read authority.

***EXCLUDE:** The user cannot access the object.

***AUTL:** The public authority of the authorization list specified in this parameter is revoked for the object. The public authority for the object becomes *EXCLUDE.

Select up to seven of the following:

***OBJEXIST:** Object existence authority provides the authority to control the object's existence and ownership. These authorities are necessary for users who want to delete the object, free storage of the object, perform save and restore operations for the object, or transfer ownership of an object. If a user has special save system authority (*SAVSYS), object existence authority is not needed. Object existence authority is required to create an object that has been named by an authority holder.

***OBJMGT:** Object management authority provides the authority to specify the security for the object, move or rename the object, and add members to database files.

***OBJOPR:** Object operational authority provides authority to look at the description of an object and use the object as determined by the user's data authorities to that object.

***ADD:** Gives the authority to add entries to an object (for example, job entries to a queue or records to a file).

***DLT:** Delete authority allows the user to remove entries from an object, for example, remove messages from a message queue or records from a file.

***READ:** Read authority provides the authority needed to show the contents of an entry in the object or to run a program.

***UPD:** Update authority provides the authority needed to change the entries in the object.

AUTL

Specifies that the authorization list is revoked from the object specified in the OBJ parameter. If public authority in the object is *AUTL, it is changed to *EXCLUDE. The authorization list's authority is then removed.

Either this parameter or the USER parameter must be specified. If this parameter is specified, the AUT parameter is ignored.

Examples

Example 1: Removing Authority From All Users Except Program Owner

```
RVKOBJAUT  OBJ(ARLIB/PROG1)  OBJTYPE(*PGM)  USER(*ALL)
```

This command removes the authorities (AUT was not specified; *CHANGE is assumed) from all users who were either explicitly or publicly authorized, except the owner, for the program (*PGM) named PROG1 located in the library named ARLIB.

Example 2: Removing Object Owner's Authority to Delete a Program

```
RVKOBJAUT  OBJ(TSMITHPGM/MITHLIB)
  OBJTYPE(*PGM)  USER(TMSMITH)
  AUT(*OBJEXIST)
```

This command removes the object owner's (TMSMITH) authority to delete a program (TSMITHPGM) in his library (SMITHLIB). The object owner might do this to ensure that the object is not deleted by mistake. If the owner ever wants to delete the object, object existence authority for the object can be granted by using the Grant Object Authority (GRTOBJAUT) command).

Example 3: Removing *DLT and *UPD Authorities

```
RVKOBJAUT   OBJ(FILEX)  OBJTYPE(*FILE)
  USER(HEANDERSON)  AUT(*DLT *UPD)
```

This command removes delete and update authorities for the file named FILEX from the user HEANDERSON.

Example 4: Removing *OBJEXIST Authority

```
RVKOBJAUT  OBJ(ARLIB/ARJOBD)  OBJTYPE(*JOBD)
  USER(RLJOHNSON)  AUT(*OBJEXIST)
```

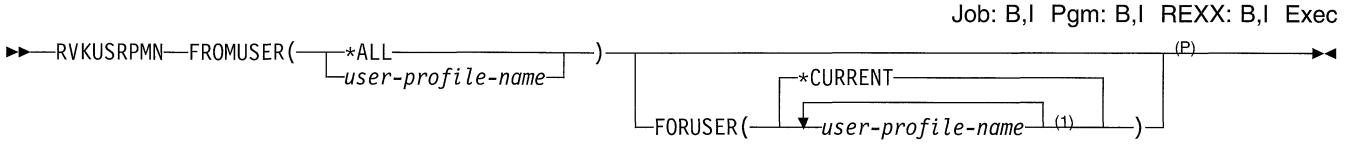
This command removes the object existence authority for the object named ARJOBD from the user RLJOHNSON. ARJOBD is a job description that is located in the library named ARLIB.

Example 5: Removing Specific Authorities

```
RVKOBJAUT  OBJ(FILEX)  OBJTYPE(*FILE)
  AUTL(FILEUSERS)
```

This command removes specific authorities for the file named FILEX from the users in the authorization list FILEUSERS.

RVKUSRPMN (Revoke User Permission) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Revoke User Permission (RVKUSRPMN) command takes away from one user (or all users) the permission to access documents and folders on behalf of another user.

Note: If work is being done on behalf of another user at the time this command is run, any functions that have started are completed; however, additional functions are not accepted.

Restriction: The user must have *ALLOBJ authority to revoke document authority for other users.

Required Parameters

FROMUSER

Specifies the user profile name of the user whose permission is being revoked.

***ALL:** All users who are currently permitted to work on behalf of another user are no longer permitted.

user-profile-name: Specify the name of the user profile that is no longer permitted to work on behalf of the user specified in the FORUSER parameter.

Optional Parameters

FORUSER

Specifies the user profile name of the user on whose behalf the user specified in the FROMUSER parameter can no longer work.

***CURRENT:** The user profile under which the current job is running is used.

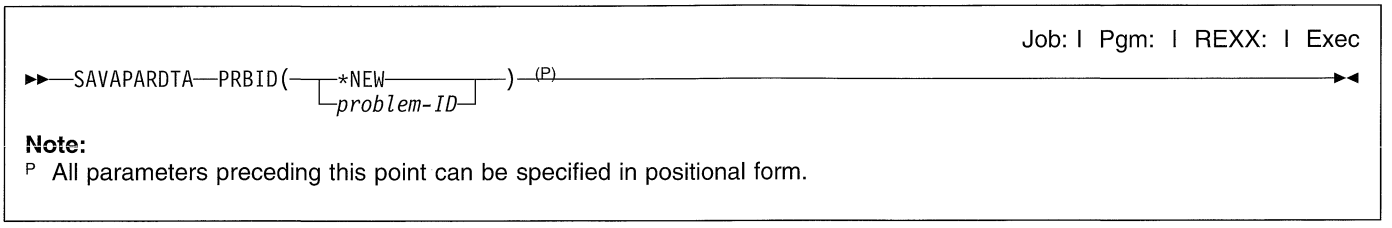
user-profile-name: Specify the name of the user profile on whose behalf other users are no longer permitted to work. Up to 50 user profile names can be specified. The user must have *ALLOBJ or *SECADM special authority to specify a user profile other than his or her own.

Example

```
RVKUSRPMN  FROMUSER(JOHNSON)  FORUSER(ANDERSON)
```

This command takes away user permission from JOHNSON for ANDERSON. The user JOHNSON is no longer allowed to work on behalf of ANDERSON.

SAVAPARDTA (Save APAR Data) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Save APAR Data (SAVAPARDTA) command allows the user to save information required for an Authorized Problem Analysis Report (APAR).

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. The following user profiles have private authorities to use the command:
   - QPGMR
   - QSYSOPR
   - QSRV
   - QSRVBAS

Required Parameters

PRBID

Specifies the identifier (ID) of the problem for which APAR data is saved.

***NEW:** An open problem log record is created to track this APAR.

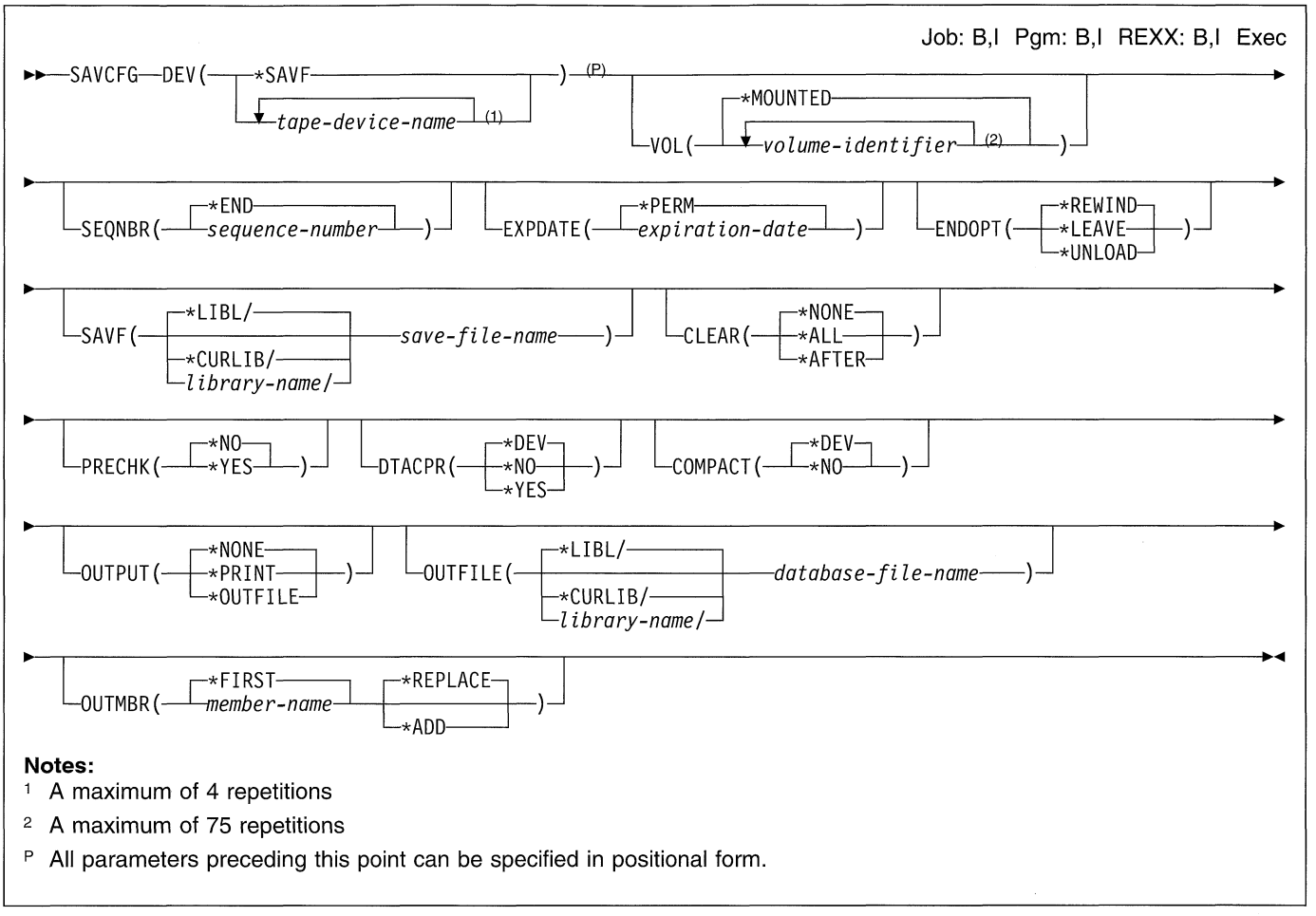
problem-ID: Specify the ID of the problem for which APAR data is saved.

Example

```
SAVAPARDTA   PRBID(*NEW)
```

This command creates an open problem log for which APAR data is saved. The user selects the data to be saved by indicating the choices on a list display. This data is saved in an APAR library.

SAVCFG (Save Configuration) Command



Notes:

1 A maximum of 4 repetitions

2 A maximum of 75 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Save Configuration (SAVCFG) command saves all configuration and system resource management (SRM) objects without requiring a system in a restricted state. The information saved includes the following:

- Line descriptions
- Controller descriptions
- Device descriptions
- Mode descriptions
- Class-of-service descriptions
- Network interface descriptions
- Connection lists
- Configuration lists
- Hardware resource data
- Token-ring adaptor data

Information saved can be restored with the RSTCFG command.

Restrictions:

1. You must have *SAVSYS special authority to use this command.
2. SAVCFG data cannot be saved to a diskette device.
3. System resource management (SRM) objects are not saved if a Work with Hardware Products (WRKHDWPRD) and Work with Hardware Resources (WRKHDWRSC) job is running at the same time.

Required Parameter

DEV

Specifies the name of the device on which the configuration information is to be saved. The device name must be known on the system by a device description.

***SAVF:** The save operation uses the save file specified on the SAVF parameter.

tape-device-name: Specify the name of one or more tape devices used for the save operation. If more than one tape device (up to four) is being used, specify the names of the devices in the order they are used. If more than one tape volume is required, the use of more

than one tape device allows one tape volume to be rewound and unloaded while the tape on another device is being processed.

Optional Parameters

VOL

Specifies the volume identifiers of the tape volumes on which the object data is to be saved. The volumes must be placed in the device in the same order as the volume identifiers are specified on this parameter.

***MOUNTED:** The data is saved on the volumes placed in the device.

volume-identifier: Specify the identifiers of one or more volumes in the order in which they are put on the device and used. Each volume identifier contains up to 6 alphanumeric characters. A blank is used as a separator character when listing multiple identifiers.

SEQNBR

Specifies the sequence number used as the starting point for the save operation when a tape is used.

***END:** The system saves the file after the last sequence number on the tape.

sequence-number: Specify the sequence number of the file.

EXPDATE

The file is permanently protected.

***PERM:** The save and restore files are permanently protected.

expiration-date: Specify the date when the file protection ends.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

SAVF

Specifies the qualified name of the save file used to contain the save data. The save file must be empty or CLEAR(*ALL) must be specified.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the save file.

CLEAR

Specifies whether uncleared tapes or save files encountered during the save operation are automatically cleared. An uncleared tape is one containing a file with an expiration date later than the date of the save operation (including files protected permanently with EXPDATE(*PERM)). This parameter does not control standard-labeling tapes; tapes must already be standard labeled.

***NONE:** None of the uncleared tapes or save files encountered during the save operation are automatically cleared. If the save operation cannot proceed because an uncleared tape is encountered, an inquiry message is sent that allows the operator either to end the save operation, or to specify that the currently selected tape be cleared so the operation can continue.

If a save file is not cleared, the inquiry message is sent to the work station message queue for an interactive job, or to the operator for a batch job. The save file should be empty, or all tapes used to perform the save operation should be cleared before the save command is issued.

***AFTER:** All the uncleared tapes after the initial tape are automatically cleared. If the operation cannot proceed because the first tape is uncleared, an inquiry message is sent asking the system operator to end the operation or to specify that the currently selected tape be cleared so the operation can continue.

***ALL:** Uncleared tapes or save files encountered during the save operation are automatically cleared.

If tapes are being used and a sequence number is specified, the tape with that sequence number is cleared, and all tapes with sequence numbers that follow the sequence number of the first tape are cleared.

PRECHK

Specifies whether the save configuration operation ends if any of the objects satisfy the following conditions:

- The objects were previously found to be damaged.
- The objects are locked by another job.
- The user does not have authority to save the objects.

***NO:** The save operation continues, saving only configuration and system resource management (SRM) objects that can be saved.

***YES:** The save operation ends before any data is written if any configuration objects or system resource manager objects cannot be saved.

DTACPR

Specifies whether data compression is performed.

***DEV:** If the tape device has the hardware compression feature installed, processing proceeds as if DTACPR(*YES) is specified. If the compression feature is not specified or if save data is written to a save file, processing proceeds as if DTACPR(*NO) is specified.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device data compression are performed if supported on the device.

***NO:** No data compression occurs.

***YES:** If the save is to tape and the target device supports compression, hardware compression is performed. If compression is not supported, or the save data is written to a save file, software compression is performed. If the save is running while other jobs on the system are active and software compression is used, overall system performance may be affected.

COMPACT

Specifies whether data compaction is performed.

***DEV:** Device data compaction is performed if the data is saved to tape and all tape devices specified on the DEV parameter support the compaction feature.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device data compression are performed if supported on the device.

***NO:** Device data compaction is not performed.

OUTPUT

Specifies whether a list with information about the saved objects is created. The information can be printed with the job's spooled output or directed to a database file.

***NONE:** No output listing is created.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: If OUTPUT(*OUTFILE) is specified, you must specify the database file name on the OUTFILE parameter.

OUTFILE

Specifies the qualified name of the database file to which the information about the object is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QASAVOBJ in QSYS with the format name QSRSAV as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which the output of the command is directed.

OUTMBR

Specifies the name of the database file member to which the output of the command is directed when *OUTFILE is specified on the OUTPUT parameter.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that receives the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it. If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

Examples

Example 1: Saving Objects

```
SAVCFG  DEV(TAP01)  CLEAR(*ALL)
```

This command saves system resource management objects (hardware resource data and token-ring adaptor data) and all configuration objects (including all line, controller, device, mode, class-of-service, and network descriptions, configuration lists, and connection lists). They are saved on the

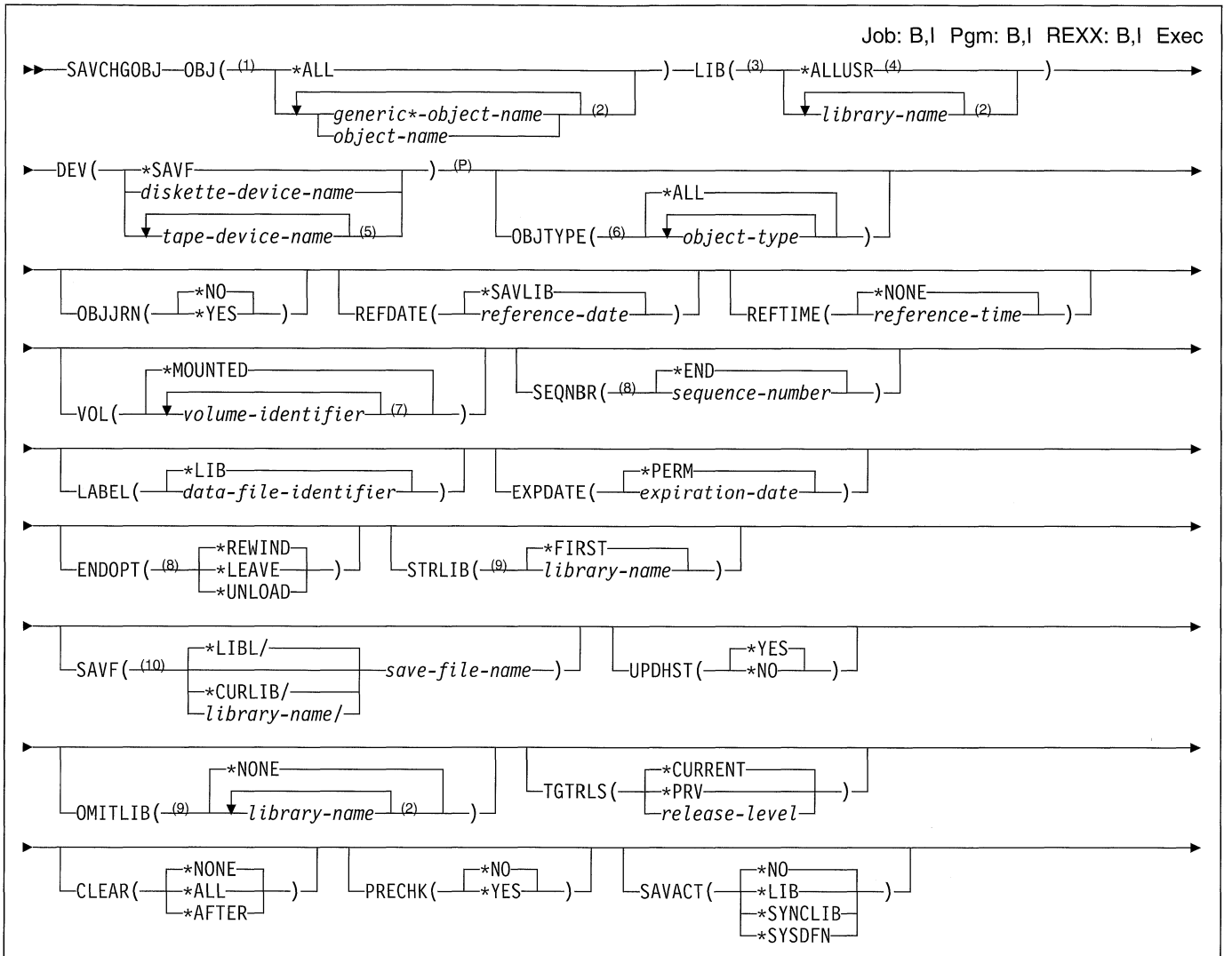
TAP01 tape drive. CLEAR(*ALL) automatically clears all uncleared tapes when they are encountered.

Example 2: Saving Objects from a Specific Tape

```
SAVCFG  DEV(TAP01)  VOL(ABC)
```

This command saves the SRM and configuration objects on the TAP01 tape drive, starting on the tape volume labeled ABC. If the save operation exceeds the storage capacity of one tape, a message requesting that another volume be put on the TAP01 tape drive is issued.

SAVCHGOBJ (Save Changed Object) Command



Notes:

1 OBJ(*ALL) is required when more than one library or LIB(*ALLUSR) is specified.

2 A maximum of 300 repetitions

3 Only one library can be specified when SAVF is used.

4 LIB (*ALLUSR) is not valid when SAVACT(*SYNCLIB) is specified.

5 A maximum of 4 repetitions

6 A list of the valid OS/400 object types for this command is in Appendix A.

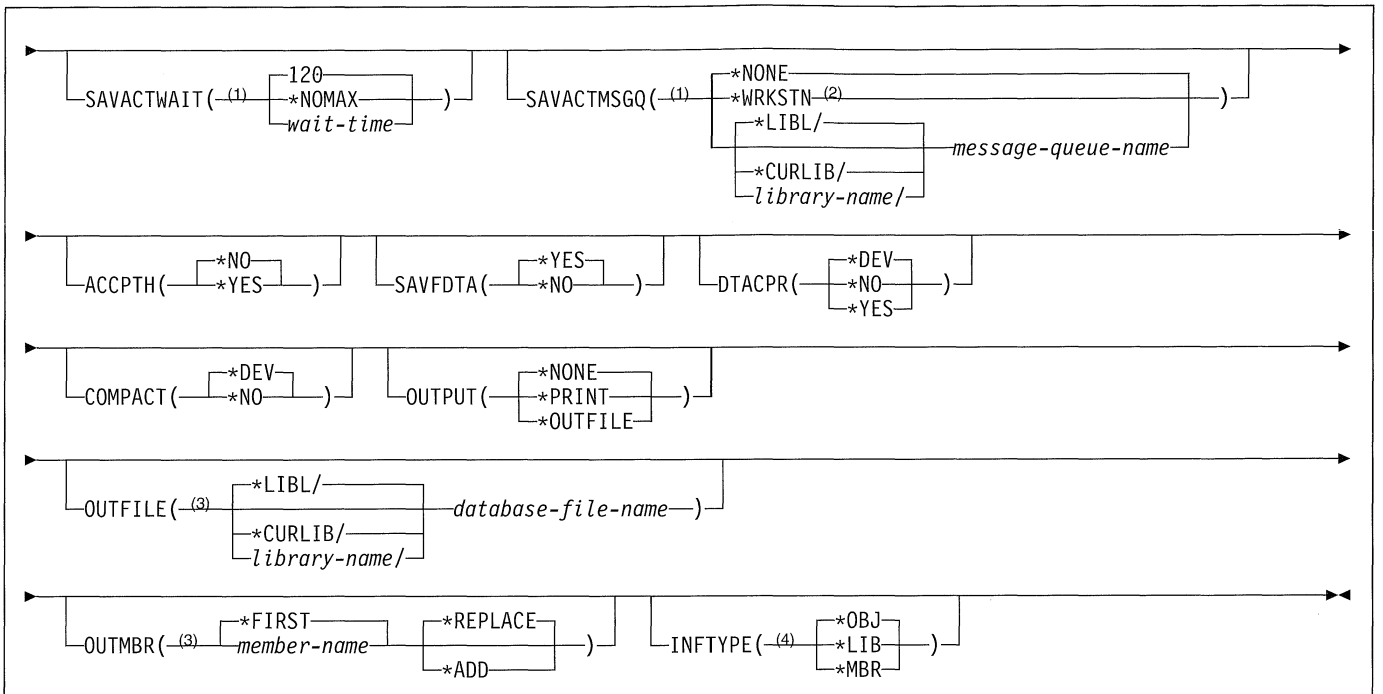
7 A maximum of 75 repetitions

8 Applies to tape devices only

9 This parameter is valid only when LIB(*ALLUSR) is specified.

10 SAVF is a required parameter when DEV(*SAVF) is specified.

P All parameters preceding this point can be specified in positional form.



Notes:

1 This parameter is valid only when SAVACT(*LIB), SAVACT(*SYNCLIB), or SAVACT(*SYSDFN) is specified.

2 This option is only valid for interactive jobs.

3 This parameter is valid only if OUTPUT(*OUTFILE) is specified.

4 This parameter is valid only if OUTPUT(*PRINT) or OUTPUT(*OUTFILE) is specified.

Purpose

The Save Changed Objects (SAVCHGOBJ) command saves a copy of each changed object or group of objects located in the same library. When OBJ(*ALL) is specified, objects can be saved from all user libraries or from up to 300 specified libraries. When saving to a save file, only one library can be specified. For database files, only the changed members are saved. Objects or members changed since the specified date and time are saved.

Objects changed since the specified date and time are saved with the following exceptions:

- If OBJJRN(*NO) is specified, database files currently being journaled are not saved, unless journaling was started after the specified date and time. This ensures that changes made to a physical file before journaling starts are not lost (because they were not journaled in a journal receiver).
- Freed objects (programs, files, journal receivers, and so forth) are not saved.
- User-defined messages, job and output queue definitions, logical file definitions, and data queue descriptions are saved, but the contents of those objects are not saved. Logical file access paths are saved if ACCPTH(*YES) is specified.

Specified objects that were changed and the libraries where they reside remain locked during the save operation.

Saved objects can be restored with the Restore Object (RSTOBJ) command.

To determine the date and time that an object was changed, run the Display Object Description (DSPOBJD) command with DETAIL(*FULL) specified. For database file members that were changed, run the Display File Description (DSPFD) command.

The types of objects that can be saved by this command are listed for the OBJTYPE parameter in "Appendix A, Expanded Parameter Descriptions." The system saves the changed objects by writing a copy of each one on diskettes, tapes, or a save file. The description of each object is changed with the date, time, and place when it was last saved. During saving to a save file, the UPDHST parameter controls this function.

Note: This command ignores all file overrides currently in effect for the job except for the save output file.

Restrictions:

1. To use this command, the user must have the special authority *SAVSYS specified in the user profile by the SPCAUT parameter. Otherwise, the user must have object existence authority for each object specified and read authority for the specified library. If the user does

not have the necessary authority to a specified object, all changed objects except that object are saved.

2. When saving to tape or diskette, the user must have use authority to the device description. When saving to a save file, the user must have object operational and add authorities to the save file.
3. All diskettes used to save the changed objects must be initialized in the save/restore format.
4. If tape is used, a standard labeled volume designation must be used.
5. No changed object that is being saved can be changed by a job that is running when the save operation occurs unless save-while-active is used.
6. When the contents of a save file are being saved to the same save file by specifying SAVFDTA(*YES), only the description of the save file is saved.
7. When the contents of a save file are saved with SAVFDTA(*YES), the save file must be restored before objects contained in it can be restored.

Required Parameters

OBJ

Specifies the names of one or more objects, or the generic names of each group of objects, to check for changes and then to save those objects that changed. All of the objects must be in the library specified on the LIB parameter.

If the OBJTYPE parameter is not specified, all the object types listed in the description of the OBJTYPE parameter are saved, provided they are in the specified library and have the specified names.

***ALL:** All changed objects in the specified libraries are saved, depending on the values specified for the OBJTYPE parameter.

generic-object-name:* Specify a generic name. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk (*) substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix, for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

object-name: Specify the names of specific objects to save. Both generic names and specific names can be specified in the same command.

LIB

Specifies the name of the library that contains the changed objects to be saved.

***ALLUSR:** Searches all user libraries. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DSULIB     #SEULIB
#COBLIB     #RPGLIB
#DFULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered "user libraries", and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be saved. Up to 300 library names can be specified. Only one library can be specified when saving to a save file.

DEV

Specifies the name of the device on which the changed objects are saved. The device name must already be known on the system by a device description.

***SAVF:** The save operation is done using the save file specified by the save file (SAVF) parameter.

diskette-device-name: Specify the name of the diskette device used to save the objects.

tape-device-name: Specify the names of one or more tape devices used for the save operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. Using more than one tape device permits one tape volume to be rewound and unloaded while another tape device processes the next tape volume.

Optional Parameters

OBJTYPE

Specifies the types of OS/400 system objects whose changes are saved. The object types saved are also the ones saved and restored by the Save Library (SAVLIB), Restore Object (RSTOBJ), and Restore Library (RSTLIB) commands. Data dictionaries and the associated files are saved only by using the SAVLIB command. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ALL:** Changes to all object types that are specified by name and that are in the specified library are saved. If *ALL is also specified on the OBJ parameter, changes to all objects in the libraries that are of the types shown in the list referred to are saved.

object-type: Specify the value for each of the types of objects that are saved, such as command (*CMD), file (*FILE), or program (*PGM).

OBJJRN

Specifies whether changes to objects currently being journaled (as specified in the Start Journal Physical File (STRJRNPF) command) are saved.

***NO:** Objects being journaled are not saved. If the STRJRNPF command was run after the specified date and time, all changed members are saved. The date and time of the last journal start operation can be shown by using the Display File Description (DSPFD) command.

***YES:** Objects whose changes are entered in the journal are saved.

REFDATE

Specifies the reference date. Objects in the specified libraries that have been changed since this date, and since the time specified in the REFTIME parameter, are saved.

***SAVLIB:** Objects changed since the last running of the Save Library (SAVLIB) command are saved. If the specified library was never saved, a message is issued and the library is not saved, but the operation continues.

reference-date: Specify the reference date. Objects that have been changed since this date are saved. If you specify a date later than the date of the running of this command, a message is issued and the operation ends. The date must be specified in the job date format.

REFTIME

Specifies the reference time. Objects in the specified libraries that have been changed at or after this time are saved.

***NONE:** No time is specified. Objects changed since the date specified on the REFDATE parameter are saved.

reference-time: Specify a time. Objects that have been changed since this time on the specified date are saved. If REFDATE(*SAVLIB) is specified, no reference time can be specified. The time is specified in 24-hour format with or without a time separator as follows:

- With a time separator, specify a string of 5 or 8 digits where the time separator separates the hours, minutes, and seconds. If this command is entered from the command line, the string must be enclosed in apostrophes. If a time separator other than the separator specified for your job is used, this command fails.
- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where **hh** = hours, **mm** = minutes, and **ss** = seconds. Valid values for **hh** range from 00 through 23. Valid values for **mm** and **ss** range from 00 through 59.

VOL

Specifies the volume identifiers of the volumes on which the data is saved. The volumes must be placed in the device in the order specified on this parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The volume currently placed in the device is used.

volume-identifier: Specify the identifiers of one or more volumes in the order in which they are put on the device and used. Each volume identifier contains up to 6 alphanumeric characters. A blank is used as a separator character when listing multiple identifiers.

SEQNBR

Specifies, when tape is used, which sequence number is used as the starting point for saving changed objects.

***END:** The system saves the changed objects after the last sequence number on the tape. If the tape is full, an error message is issued and the operation ends.

sequence-number: Specify the sequence number of the file.

LABEL

Specifies the name that identifies the data file on the tape or diskette that is used for the save operation. If the LABEL parameter is used on the save command, the same label must be specified on the restore command.

***LIB:** The file label is created by the system using the name of the library specified on the LIB parameter.

data-file-identifier: Specify the data file identifier (a maximum of 17 characters) of the data file used for the save operation. This option is valid only for the single-library save operations. Do not specify *SAVLIB; the system will return an error message if *SAVLIB is specified.

EXPDATE

Specifies the expiration date. The files cannot be overwritten until the expiration date. The expiration date must be later than or equal to the current date.

***PERM:** The file is permanently protected.

expiration-date: Specify the date when protection for the file ends.

Note: For operations saving objects to diskette, the expiration date specified must be later than the date of the save operation. Otherwise, the unprotected save/restore files may be lost when the next save/restore file is written during the save operation.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

STRLIB

Specifies the library with which to begin the *ALLUSR save. This parameter is used to recover from ended or failed *ALLUSR save operations.

Note: The user should not attempt to specify LIB(*ALLUSR) on a failed tape. The proper recovery procedure is to start over with a new tape, as if the user is starting the complete save operation again.

***FIRST:** The save operation begins with the first library in alphabetical order.

library-name: Specify the name of the library with which to begin the save.

SAVF

Specifies the qualified name of the save file used to contain the save data. The save file must be empty or CLEAR(*ALL) must be specified.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the save file used to contain the saved data.

UPDHST

Specifies whether the save history information of each saved object is changed with the date, time, and location of this save operation. The save history information for an object can be displayed by using the Display Object Description (DSPOBJD) command. This information is used to determine which journal entries are processed when RCVRNG(*LASTSAVE) and FROMENT(*LASTSAVE) are used on the Apply Journaled Changes (APYJRNCHG) command.

Note: This parameter applies only when saving to a save file by using the SAVF parameter.

***YES:** The last save date, time, and location information is updated in each object saved. This value works the same way as a save operation to diskette or tape.

***NO:** The save history information contained in the description of each object saved is not updated.

Note: UPDHST(*NO) should only be used for a save operation that is not intended for recovery or saving. For example, if the save data is transmitted, record by record, to another system and the save file then deleted, the user probably does not want to update the save history information.

OMITLIB

Specifies up to 300 libraries to be excluded from the save operation. This parameter is valid only if LIB(*ALLUSR) is specified.

***NONE:** No libraries are excluded from the save operation.

library-name: Specify the name of the library to be excluded from the save operation. Up to 300 library names can be specified.

TGTRLS

Specifies the release level of the operating system on which you intend to restore and use the object.

The release level is specified in the format VxRxMx, where Vx is the version, Rx is the release, and Mx is the modification level. The release level V2R3M0 is Version 2, Release 3, Modification 0.

To specify that an object be saved for distribution to a system at a different release level than the system on which the save operation is to occur, the procedure differs for program and non-program objects and by the release level on which program objects are created. If, for example, you are saving an object for distribution to a target system running on an earlier release, you have the following choices:

For program objects

- If the program was created at a release level more current than the targeted earlier release, you must (1) create the program again specifying the targeted earlier release, (2) save the program specifying the targeted earlier release, and then (3) restore the program on the target system.
- If the program was created at the same release level as the target system, you can (1) save the program specifying the targeted earlier release and then (2) restore the program on the target system.

For non-program objects

You can (1) save the object specifying the targeted earlier release and then (2) restore the object on the target system.

Note: Not all objects can be targeted to another release. To find out which objects are supported, see the chart in the *Basic Backup and Recovery Guide*.

***CURRENT:** The object is to be restored on the release of the operating system currently running on your system. If V2R3M0 is running on the system, *CURRENT means that you intend to restore the object on a system with V2R3M0 installed. You also can restore the object on a system with any later release of the operating system installed.

***PRV:** The object is to be restored on the previous release with modification level 0 of the operating system.

If V2R3M0 is running on the system, *PRV means that you intend to restore the object on a system with V2R2M0 installed. You also can restore the object on a system with any later release of the operating system installed.

release-level: Specify the release level in the format VxRxMx. The object can be restored on a system with the specified release or with any later release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. Valid values for release V2R3M0 are: V2R3M0 and V2R2M0.

You can use the following table to find out what values can be specified and what release is specified when you choose the *CURRENT and *PRV values on your operating system.

Table 71. Valid Values for TGTRLS Parameter in VxRxMx Format

Your OS/400 System Release	*CURRENT Value	*PRV Value	*Release-Level* Variables
V2R3M0	V2R3M0	V2R2M0	V2R3M0 V2R2M0
V2R2M0	V2R2M0	V2R1M0	V2R2M0 V2R1M1 V2R1M0
V2R1M1	V2R1M1	V1R3M0	V2R1M1 V2R1M0 V1R3M0
V2R1M0	V2R1M0	V1R3M0	N/A
V1R3M0	V1R3M0	V1R2M0	N/A
V1R2M0	V1R2M0	V1R1M0 V1R1M2	N/A

CLEAR

Specifies whether tapes, diskettes, or save files that contain active data and are encountered during the save operation are automatically cleared. An uncleared tape or diskette is one containing a file with an expiration date later than the date of the save operation (including files protected permanently with EXPDATE(*PERM)). This parameter does not initialize diskettes or assign standard labels to tapes; diskettes must already be initialized in the save/restore format and tapes must be standard labeled.

***NONE:** None of the uncleared tapes, diskettes, or save files encountered during the save operation are automatically cleared. If the save operation cannot proceed because an uncleared tape, diskette, or save file is encountered, an inquiry message is sent to the operator. This allows ending the save operation or specifying that the currently selected tape, diskette, or save file be cleared so the operation can continue. If a save file is not cleared, the inquiry message is sent to the work station message queue for an interactive job, or to the operator for a batch job. The save file must be empty, or all tapes or diskettes used to perform the save operation should be cleared before the save command is issued.

***ALL:** All the uncleared tapes, diskettes, or save files encountered during the save operation are automatically cleared. If tapes are used and a sequence number is specified, the tape is cleared and, starting with that sequence number, all tapes following the first tape are cleared.

***AFTER:** All the uncleared tapes or diskettes after the initial tape or diskette are automatically cleared. This option is not valid for save/restore operations to save files. If the operation cannot proceed because the first tape or diskette is uncleared, an inquiry message is sent to the system operator. This allows ending the operation or specifying that the currently selected tape or diskette be cleared so the operation can continue.

PRECHK

Specifies whether the save operation for a library ends if all objects specified by this command do not satisfy the following conditions:

- The objects exist
- The objects were not previously found to be damaged
- The objects are not locked by another job
- The requester of the save operation has authority to save the objects

***NO:** The save operation for a library continues, saving only objects that can be saved.

***YES:** Ends the save operation for a library before any data is written if, after all specified objects are checked, one or more objects cannot be saved. If several libraries are specified, the save operation continues with the next library. However, if PRECHK(*YES) and SAVACT(*SYNCLIB) are specified and one or more objects in any library to be saved cannot be saved, the save operation ends and no objects are saved.

SAVACT

Specifies whether an object can be updated while it is being saved.

Note: If your system is in a restricted state and the SAVACT parameter is specified, the save operation is performed as if SAVACT(*NO) was specified.

***NO:** Objects that are in use are not saved. Objects cannot be updated while being saved.

***LIB:** Objects in a library can be saved while they are in use by another job. All of the objects in a library reach a checkpoint together and are saved in a consistent state in relationship to each other.

Note: Libraries with thousands of objects may be too large for this option.

***SYNCLIB:** Objects in a library can be saved while they are in use by another job. All of the objects and all of the libraries in the save operation reach a checkpoint

together and are saved in a consistent state in relationship to each other.

***SYSDFN:** Objects in a library can be saved while they are in use by another job. Objects in a library may reach checkpoints at different times and may not be in a consistent state in relationship to each other.

Note: Specifying this value eliminates some size restrictions and may enable a library to be saved that could not be saved with SAVACT(*LIB).

SAVACTWAIT
Specifies the amount of time to wait for a commit boundary or an object that is not available, before continuing the save. If an object remains in use for the specified time, the object is not saved. If a commit boundary is not reached in the specified time, the save operation is ended.

120: The system waits up to 120 seconds for a commit boundary or an object before continuing the save operation.

***NOMAX:** No maximum wait time exists.

wait-time: Specify the time (in seconds) to wait for a commit boundary or an object before continuing the save operation.

SAVACTMSGQ
Specifies the message queue that the save operation uses to notify the user that the checkpoint processing for a library is complete. A separate message is sent for each library to be saved when SAVACT(*SYSDFN) or SAVACT(*LIB) is specified. When SAVACT(*SYNCLIB) is specified, one message is sent for all libraries in the save operation.

This parameter can be used to save the objects at a known, consistent boundary to avoid additional recovery procedures following a restore operation. Applications can be stopped until the checkpoint processing complete message is received.

***NONE:** No notification message is sent.

***WRKSTN:** The notification message is sent to the work station message queue.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue.

ACCPTH
Specifies whether the logical file access paths that are dependent on the physical files being saved are also saved. The access paths are saved only if all members on which the access paths are built are included in this save operation. Informational messages are sent indicating the number of logical file access paths saved with each physical file. All physical files on which an access path is built must be in the same library. This parameter does not save logical file objects; it controls only the saving of the access paths. More information on the restoring of saved access paths is in the *Basic Backup and Recovery Guide*.

Attention!

If the based-on physical files and the logical files are in different libraries, the access paths are saved.

However, if the logical files and the based-on physical files are in different libraries and the logical files or physical files do not exist at restore time (such as during disaster recovery or the files were deleted) the access paths are not restored. They are rebuilt.

For the fastest possible restore operation for logical files, the logical files and the based-on physical files must be in the same library and must be saved at the same time.

***NO:** Only objects specified on the command are saved. No logical file access paths are saved.

***YES:** The specified physical files and all eligible logical file access paths over them are saved.

SAVFDTA
Specifies, for save file objects, whether the description of a save file, or both the description and the contents of a save file, are saved on a tape, diskette, or another save file.

***YES:** The description and contents of the save file are saved.

***NO:** Only the description of a save file is saved.

DTACPR
Specifies whether data compression is performed.

***DEV:** If the tape device has the hardware compression feature installed, processing proceeds as if DTACPR(*YES) is specified. If the compression feature is not installed or if save data is written to a diskette or save file, processing proceeds as if DTACPR(*NO) is specified.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device

data compression are performed if supported on the device.

***NO:** No data compression or decompression occurs.

***YES:** If the save operation is to tape and the target device has the hardware compression feature, hardware compression is performed. If the feature is not present, or if the save data is written to a diskette or save file, software compression is performed. If the save operation is being done while other jobs on the system are active and software compression is used, the overall system performance may be affected.

COMPACT

Specifies whether device data compaction is performed.

***DEV:** Device data compaction is performed if the data is saved to tape and all tape devices specified on the DEV parameter support the compaction feature.

***NO:** Device data compaction is not performed.

OUTPUT

Specifies whether a list with information about the saved objects is created. The information can be printed with the job's spooled output or directed to a database file. The output or spooled print file does not include records for objects not changed.

***NONE:** No output listing is created.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: If OUTPUT(*OUTFILE) is specified, you must specify the database file name on the OUTFILE parameter.

OUTFILE

Specifies the qualified name of the database file to which the information about the object is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QASAVOBJ in QSYS with the format name QSRSAV as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which the output of the command is directed.

OUTMBR

Specifies the name of the database file member to which the output of the command is directed when OUTPUT(*OUTFILE) is specified.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that receives the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it. If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

INFTYPE

Specifies the type of information printed or directed to the database file.

***OBJ:** The list contains an entry for each object requested to be saved.

***LIB:** The list contains an entry for each library requested to be saved.

***MBR:** The list contains an entry for each object or, for database files, each member requested to be saved.

Examples

Example 1: Saving Changed Files

```
SAVCHGOBJ  OBJ(ORD*)  LIB(DSTPRODLB)  DEV(TAP01)
  OBJTYPE(*FILE)  REFDATE(122290)
```

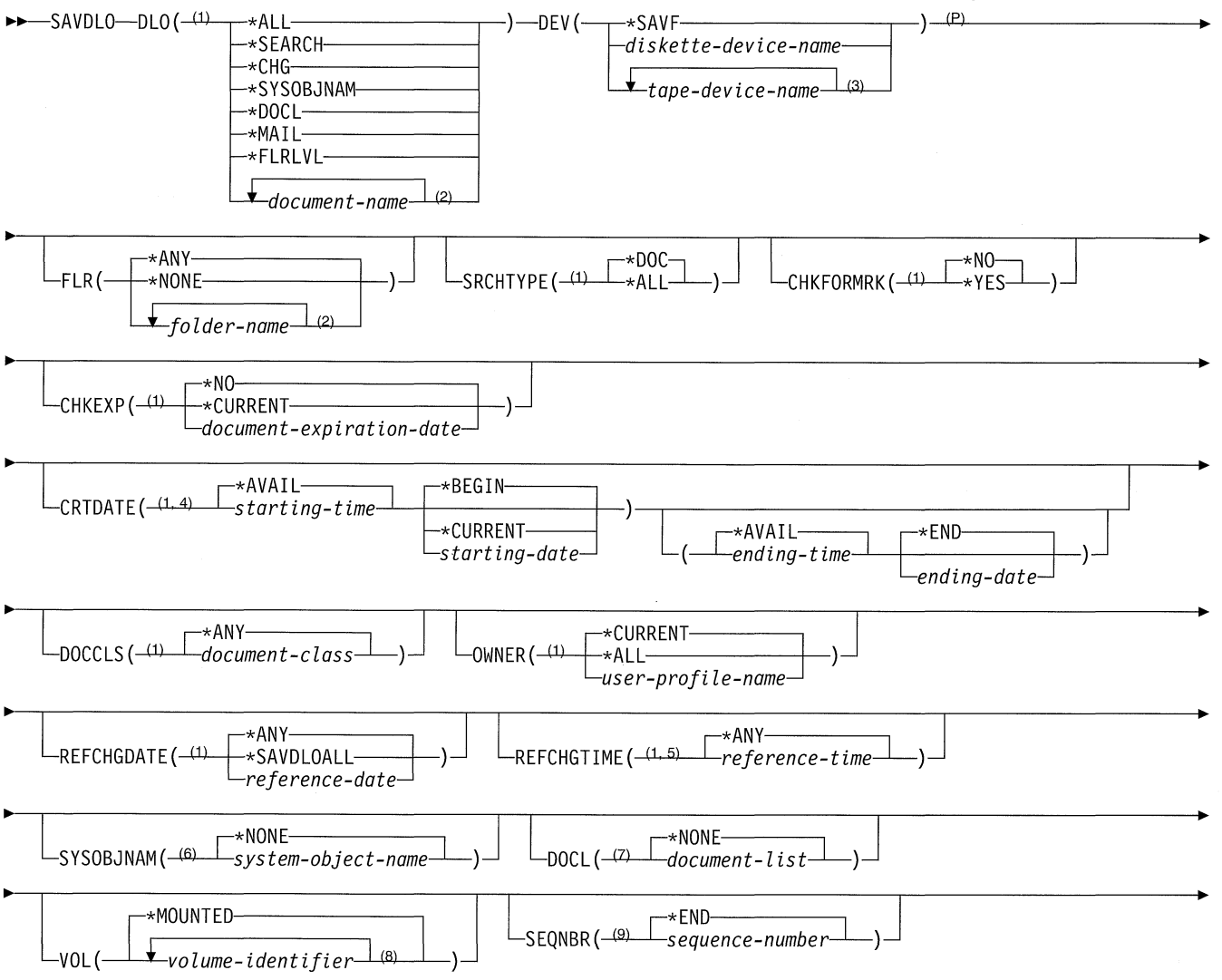
This command saves all files with names that start with the characters ORD in the library named DSTPRODLB that were changed since December 22, 1990.

Example 2: Saving Description and Data for Save Files

```
SAVCHGOBJ  OBJ(FILE*)  LIB(MYLIB)  DEV(TAP01)
  OBJTYPE(*FILE)  REFDATE(122290)  SAVFDTA(*YES)
```

This command saves all files with names that start with the characters FILE* in the library named MYLIB that were changed since December 22, 1990. It also saves the description and the data for all save files that match this selection criteria.

SAVDLO (Save Document Library Object) Command



Notes:

1 CHKFORMRK, CHKEXP, CRTDATE, DOCCLS, OWNER, REFCHGDATE, REFCHGTIME, and SRCHTYPE are allowed only if DLO(*SEARCH) is specified.

2 A maximum of 300 repetitions

3 A maximum of 4 repetitions

4 CRTDATE contains two lists of two elements each.

5 This parameter cannot be specified if REFCHGDATE(*SAVDLOALL) is specified.

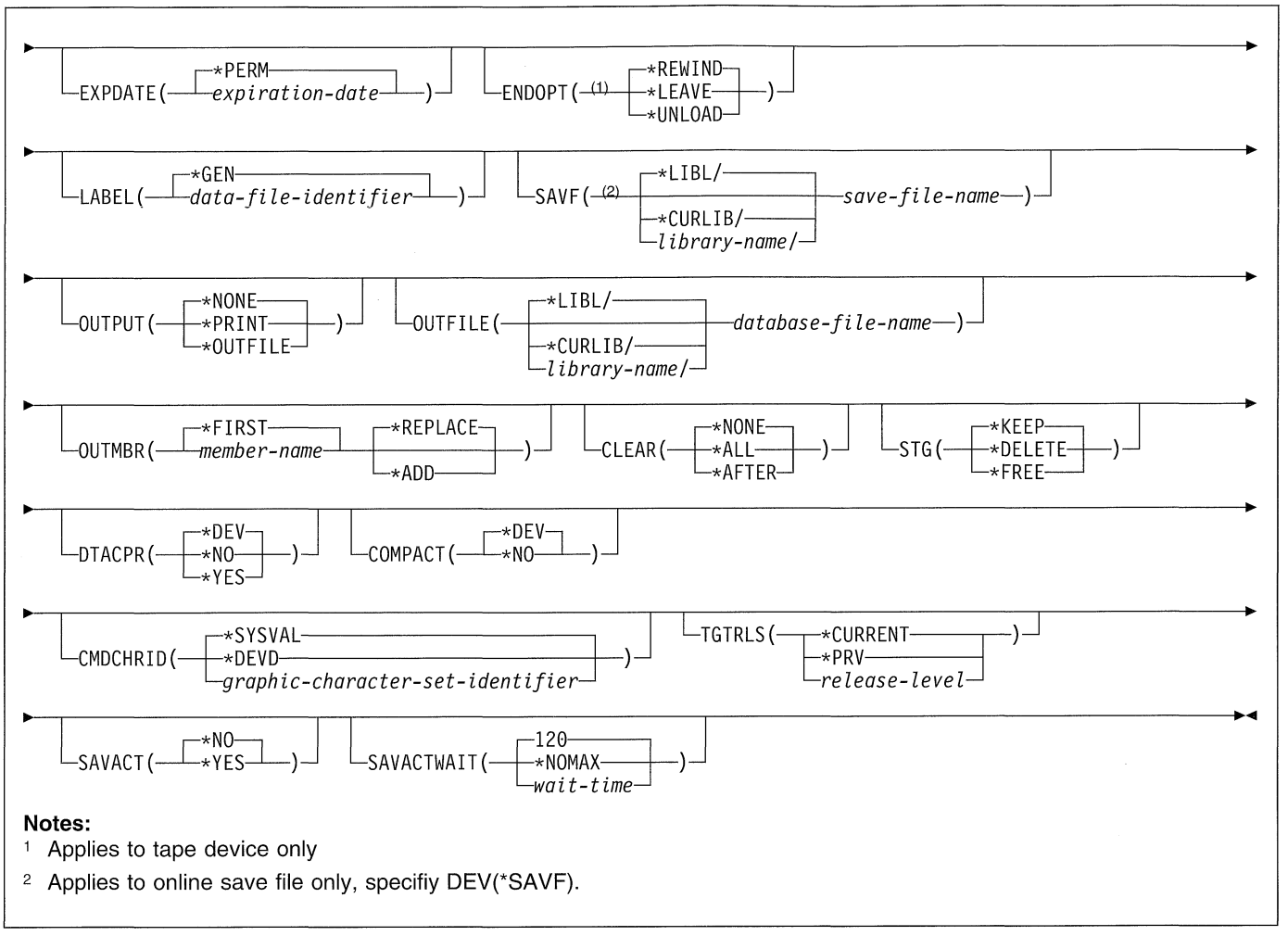
6 SYSOBJNAM required if DLO(*SYSOBJNAM) is specified.

7 DOCL required if DLO(*DOCL) is specified.

8 A maximum of 75 repetitions

9 Applies to tape device only

P All parameters preceding this point can be specified in positional form.



Notes:
1 Applies to tape device only
2 Applies to online save file only, specifiy DEV(*SAVF).

Purpose

The Save Document Library Object (SAVDLO) command saves copies of the following:

- Documents
- Folders
- Distribution objects (mail)

Notes:

1. When a folder is saved, the folder object is saved along with the documents contained in that folder and the subfolders and documents in the subfolders and all successively nested folders and documents. Specific folders can be saved individually using DLO(*FLRLVL).
2. Distribution objects (mail) cannot be saved or restored for individual users. Mail can be saved only for all users.
3. SAVDLO does not require a dedicated system; however, individual objects in use when the save is issued cannot be saved. To ensure all document library objects are saved, run this command when no office activity is occurring on the system.

Restrictions:

1. You must have *ALLOBJ or *SAVSYS special authority to use the following parameter combinations on this command:
   - DLO(*ALL) FLR(*ANY)
   - DLO(*CHG)
   - DLO(*MAIL)
   - DLO(*SEARCH) OWNER(*ALL)
   - DLO(*SEARCH) OWNER(*user-profile-name*)

   where the *user profile name* specified is not the *user profile name* of the user issuing the SAVDLO command.
2. Users that do not have *ALLOBJ or *SAVSYS special authority must:
   - Have *ALL authority for each document or folder to be saved
   - Be enrolled as Document Interchange Architecture (DIA) users
3. This command cannot be used while another job is running commands such as RCLDLO, SAVDLO, and RSTDLO because exclusive use of internal objects may have been obtained by these commands.
4. Determining document or folder ownership does not include checking group profiles if one is associated with the specified user profile.

Required Parameters

DLO

Specifies the document library objects to save.

***ALL:** All document library objects further qualified by the FLR parameter are to be saved. Specifying DLO(*ALL) FLR(*ANY) saves all document library objects.

***SEARCH:** All documents and folders that meet the specified search criteria are saved. Search criteria are specified by using the following parameters: FLR, CHKFORMRK, CHKEXP, CRTDATE, DOCCLS, OWNER, REFCHGDATE, and REFCHGTIME.

***CHG:** All documents created or changed, all folders created since the last complete save operation, and all mail is saved.

***SYSOBJNAM:** The documents with system object names specified on the SYSOBJNAM parameter are saved.

***DOCL:** The list of documents referred to in a document list specified on the DOCL parameter are saved.

***MAIL:** The distribution objects and documents referred to by a mail log are saved.

***FLRLVL:** The folders specified on the FLR parameter and the documents in the folders are saved. Subfolders are not saved.

document-name: Specify the user-assigned names of the documents that are to be saved. A maximum of 300 documents can be specified. All documents specified must be in the same folder and that folder must be specified on the FLR parameter.

DEV

Specifies the name of the device used for the save operation. The device name must already be known on the system by a device description.

***SAVF:** The save file specified on the SAVF parameter is to be used.

diskette-device-name: Specify the name of the diskette device to be used.

tape-device-name: Specify the names of one or more tape devices used for the save operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. Using more than one tape device permits one tape volume to be rewound and unloaded while another tape device processes the next tape volume.

Optional Parameters

FLR

Specifies the name of the folder to save.

***ANY:** Document library objects can be saved from any folder. Consider the following when using the FLR parameter:

- FLR(*ANY) is not valid when one of the following is specified:
  - DLO(*DOCL)
  - DLO(*FLRLVL)
  - DLO(*document-name*)
- FLR(*ANY) is required when one of the following is specified:
  - DLO(*CHG)
  - DLO(*SYSOBJNAM)
  - DLO(*MAIL)
  - DLO(*SEARCH) SRCHTYPE(*ALL)
- When SAVDLO DLO(*ALL) FLR(*ANY) is specified, the following are saved:
  - All documents
  - All folders
  - All distribution objects (mail)

***NONE:** The documents saved are not in any folder. FLR(*NONE) is valid only when one of the following is specified:

- DLO(*ALL)
- DLO(*SEARCH) SRCHTYPE(*DOC)

folder-name: Specify the user-assigned name of the folder from which the documents are to be saved. The folder name can be a maximum of 63 characters in length. A maximum of 300 folders can be specified.

- Folder objects specified here are saved only when DLO(*ALL) or DLO(*FLRLVL) is specified.
- FLR(*folder-name*) is not valid when one of the following is specified:
  - DLO(*SYSOBJNAM)
  - DLO(*MAIL)
  - DLO(*SEARCH) SRCHTYPE(*ALL)
- Only one folder name can be specified when one of the following is specified:
  - DLO(*DOCL)
  - DLO(*SEARCH) SRCHTYPE(*DOC)
  - DLO(*document-name*)

SRCHTYPE

Specifies the type of objects for which to search. This parameter is valid only when DLO(*SEARCH) is specified.

***DOC:** Only documents are to be searched and saved.

***ALL:** Documents and folders are to be searched and saved.

CHKFORMRK

Specifies whether documents that have been marked for offline storage are saved. This parameter is valid only when DLO(*SEARCH) SRCHTYPE(*DOC) is specified.

***NO:** All documents that meet the other search values for this save operation are to be saved regardless of whether or not they were marked for offline storage.

***YES:** Only those documents that meet the other search values and are marked for offline storage are saved. Documents may be marked:

- Keep
- Free
- Delete

CHKEXP

Specifies the date for which all documents with an expiration date before this date are to be saved. This date is assigned by the user when the document was created to specify when the document is no longer needed. This parameter is valid only when DLO(*SEARCH) SRCHTYPE(*DOC) is specified.

***NO:** The expiration date is ignored.

***CURRENT:** All documents with an expiration date before today's date are to be saved.

document-expiration-date: Specify a date when all documents that have an expiration date before this date are to be saved.

CRTDATE

Specifies that documents and folders that have a creation date during the time period specified are to be saved. The time period is specified by a starting time and date and an ending time and date. This parameter is valid only when DLO(*SEARCH) is specified.

Element 1: Starting Time

***AVAIL:** Documents and folders created at any time are eligible for selection.

starting-time: Specify the starting time.

When the starting time is used as a search value, the starting date must not be *BEGIN. The starting time must be the same as the value specified on the REFCHGTIME parameter when the REFCHGTIME parameter is specified.

The time is specified in 24-hour format with or without a time separator as follows:

- With a time separator, specify a string of 5 or 8 digits where the time separator separates the hours, minutes, and seconds. If this command is entered from the command line, the string must be enclosed in apostrophes. If a time separator other than the separator specified for your job is used, this command fails.
- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where **hh** = hours, **mm** = minutes, and **ss** = seconds. Valid values for **hh** range from 00 through 23. Valid values for **mm** and **ss** range from 00 through 59.

Element 2: Starting Date

***BEGIN:** Documents and folders are saved regardless of the creation date associated with the object.

***CURRENT:** The current date is used.

starting-date: Specify the starting date. The date must be specified in the job date format. The start date must be the same as the value specified on the REFCHGDATE parameter when the REFCHGDATE parameter is specified.

Element 3: Ending Time

Documents and folders must have been created before this time to be saved.

***AVAIL:** Documents and folders created at any time are saved.

ending-time: Specify the ending time. When the ending time is to be used as a search value, the ending date must not be *END. See the description of *starting-time* for details about how time can be specified.

Element 4: Ending Date

Documents and folders must have been created on or before this date to be saved.

***END:** Documents created on any date after the starting date are eligible to be saved.

ending-date: Specify the ending date in job date format.

DOCCLS

Specifies the class of documents being saved.

The class is assigned by the user when the document is created. This parameter is valid only when DLO(*SEARCH) SRCHTYPE(*DOC) is specified.

Note: Although document classes are user-assigned, double-byte character set (DBCS) data cannot be specified on this parameter.

***ANY:** The document class is not used to select documents being saved.

document-class: Specify a character string, ranging from 1 through 16 characters in length, that is used to select documents being saved.

OWNER

Specifies the owner of the documents and folders being saved. This parameter is valid only when DLO(*SEARCH) is specified.

***CURRENT:** Documents and folders owned by the user issuing this command are to be saved.

***ALL:** The OWNER parameter is not used to select documents and folders to be saved. *ALLOBJ or *SAVSYS special authority is required when OWNER(*ALL) is specified.

user-profile-name: Specify the name of the user who owns the documents and folders that are to be saved. All documents and folders owned by this user and that meet the other search values specified are to be saved.

*ALLOBJ or *SAVSYS special authority is required if the user profile specified is other than the user profile of the user issuing this command.

REFCHGDATE

Specifies the date after which the folders that are created and the documents that are changed or created are to be saved. The change date is updated when the document content or description is changed. This parameter is valid only when DLO(*SEARCH) is specified.

***ANY:** No specific date is specified. Documents are saved regardless of the date they were created or changed. Folders are saved regardless of the date they were created.

***SAVDLOALL:** Folders that have been created and documents that have been created or changed since the last complete save operation are to be saved.

reference-date: Specify the date after which the created folders or the created or changed documents are saved.

REFCHGTIME

Specifies the time, relative to the date specified on the REFCHGDATE parameter, after which the folders that are created and the documents that are changed or created are to be saved. The change time is updated when the document content or description is changed. This parameter is valid only when DLO(*SEARCH) is specified.

***ANY:** No time is specified. The documents are saved regardless of the date they were created or changed. Folders are saved regardless of the date they were created.

reference-time: Specify the time after which the created folders or the created or changed documents are saved. Information on how to specify time is in the CRTDATE parameter description.

SYSOBJNAM

Specifies the system object name. This parameter is valid only when DLO(*SYSOBJNAM) is specified.

***NONE:** The system object name is ignored.

system-object-name: Specify the system object name of the document to be saved. A full 10 characters must be specified. A maximum of 300 names can be specified.

DOCL

Specifies a document list that contains the names of documents to be saved. The document list must be in a folder and the folder must be specified on the FLR parameter. This parameter is valid only when DLO(*DOCL) is specified.

The user must have *USE authority to the document list.

Note: The document list must be the result of a local search, not a remote search.

***NONE:** Documents are not to be saved from a document list.

document-list: Specify the document list containing the names of the documents to be saved.

VOL

Specifies the volume identifiers of the volumes on which the data is saved. The volumes must be placed in the device in the order specified on this parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The data is saved on the volumes placed in the device.

volume-identifier: Specify the identifiers of one or more volumes in the order in which they will be placed on the device. A maximum of 75 volumes can be specified.

SEQNBR

Specifies the starting tape sequence number to use for saving the documents library objects.

***END:** The system saves the document library objects after the last sequence number on the tape.

sequence-number: Specify the sequence number of the file that is used as a starting point to save the document library objects.

EXPDATE

Specifies the expiration date of the diskette or tape file being created. The diskette or tape files cannot be overwritten until the expiration date. The expiration date must be later than or equal to the current date.

Notes:

1. This parameter is valid for both tape and diskette. For save operations to diskette, the expiration date specified must be later than the date of the save operation. Otherwise, the save and restore files whose expiration date has been exceeded may be lost when the next save and restore file is written during the save operation.
2. Specifying this parameter does not protect against a later save operation specifying CLEAR(*ALL).

***PERM:** The file is permanently protected.

expiration-date: Specify the date when protection for the file ends.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

Note: If no objects are saved the tape volume is not opened and the ENDOPT parameter is ignored.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape is not rewound; another save operation can start at the current position on the tape.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

LABEL

Specifies the name that identifies the data file on the tape or diskette used for the save. If the LABEL parameter is used, the label must be specified on the restore command.

***GEN:** The system generates the label name.

data-file-identifier: Specify the data file identifier that is used as the label for the data file being used for the save operation. Up to 17 characters can be specified.

SAVF

Specifies the qualified name of the save file used to contain the save data.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the save file.

OUTPUT

Specifies whether a list with information about the saved document library objects is created.

***NONE:** No output is created.

***OUTFILE:** The output is directed to a database file specified on the OUTFILE parameter. The file must have the same format as database file QAOJSAVO.

***PRINT:** The output is printed with the job's spooled output.

OUTFILE

Specifies the qualified name of the database file to which the information about the document library objects is directed. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QAOJSAVO in QSYS as a model with the format name of QJSDLO. This parameter is valid only when OUTPUT(*OUTFILE) is specified.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to be used.

OUTMBR

Specifies the name of the database file member to which the output is directed when OUTPUT(*OUTFILE) is specified.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that is to receive the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

CLEAR

Specifies whether tapes, diskettes, or save files that contain active data and are encountered during the save operation are automatically cleared. An uncleared tape or diskette is one containing a file with an expiration date later than the date of the save operation (including files protected permanently with EXPDATE(*PERM)). This parameter does not initialize diskettes or assign standard labels to tapes; diskettes must already be initialized in the save/restore format and tapes must be standard labeled.

***NONE:** Tapes, diskettes, or save files that contain active data and are encountered during the save operation are not automatically cleared. If the save operation cannot proceed because an uncleared tape, diskette, or save file is encountered, an inquiry message is sent to the operator. The operator can either end the save operation or specify that the currently selected tape, diskette, or save file be cleared so the operation can continue. If a save file contains data, an inquiry message is sent to the workstation message queue for an interactive job, or to the operator for a batch job. The operator can either end the save operation or specify that the currently selected diskette or tape be cleared so the operation can continue. A save file must be empty or all tapes or diskettes used to perform the save operation must be cleared before the command is issued.

***ALL:** Tapes, diskettes, or save files that contain active data and are found during the save operation are cleared so the save operation can continue. If tapes are used and a sequence number is specified, the tape is cleared starting with that sequence number; all tape files following the first tape file are cleared.

***AFTER:** All diskettes and tape files that contain active data and are found after the first diskette or tape file are

cleared. This option is not valid to save files. If the operation cannot proceed because the first diskette or tape is uncleared, an inquiry message is sent to the operator. The operator can either end the save operation or specify that the currently selected diskette or tape be cleared so the operation can continue.

STG

Specifies, only for filed documents, whether the system storage that is occupied by the document being saved is kept, deleted, or freed after the save operation is completed.

Note: STG(*DELETE) and STG(*FREE) are not valid when any of the following are specified:

- DLO(*ALL) FLR(*ANY)
- DLO(*SEARCH) CHKFORMRK(*YES)
- DLO(*CHG)
- DLO(*MAIL)
- SAVACT(*YES)

***KEEP:** The storage occupied by the document remains unchanged after the save operation.

***DELETE:** The document object and all search terms are deleted from the system after the save operation.

***FREE:** The document description and search terms remain on the system but the storage occupied by the document is deleted after the save operation.

DTACPR

Specifies whether data compression is performed.

***DEV:** If the tape device has the hardware compression feature installed, processing proceeds as if DTACPR(*YES) is specified. If the compression feature is not installed or if save data is written to a diskette or save file, processing proceeds as if DTACPR(*NO) is specified.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device data compression are performed if supported on the device.

***NO:** No data compression is performed.

***YES:** If the save is to tape and the target device has the hardware compression feature, hardware compression is performed. If the feature is not present or if the save data is written to diskette or save file, software compression is performed. If the save operation is run while other jobs on the system are active and software compression is used, there may be an effect on the overall system performance.

COMPACT

Specifies whether device data compaction is performed.

***DEV:** Device data compaction is performed if the data is saved to tape and all tape devices specified on the DEV parameter support the compaction feature.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device data compression are performed if supported on the device.

***NO:** Device data compaction is not performed.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the *Guide to Programming Displays*.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

graphic-character-set-identifier: Specify the graphic character set and code page values that were used to create the command parameters. Each value can be up to 3 bytes in length and must be separated by at least one blank.

TGTRLS

Specifies the release level of the operating system on which you intend to restore and use the object.

The release level is specified in the format VxRxMx, where Vx is the version, Rx is the release, and Mx is the modification level. The release level V2R3M0 is Version 2, Release 3, Modification 0.

***CURRENT:** The object is to be restored on the release of the operating system currently running on your system. If V2R3M0 is running on the system, *CURRENT means that you intend to restore the object on a system with V2R3M0 installed. You also can restore the object on a system with any later release of the operating system installed.

***PRV:** The object is to be restored on the previous release with modification level 0 of the operating system.

If V2R3M0 is running on the system, *PRV means that you intend to restore the object on a system with V2R2M0 installed. You also can restore the object on a system with any later release of the operating system installed.

release-level: Specify the release level in the format VxRxMx. The object can be restored on a system with the specified release or with any later release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. Valid values for release V2R3M0 are: V2R3M0 and V2R2M0.

SAVACT

Specifies whether an object can be updated while it is being saved.

***NO:** Objects are not allowed to be saved if they are in use by another job.

***YES:** Objects are allowed to be changed during the save request. Objects that are in use but are not using application recovery will not be saved. See the *Advanced Backup and Recovery Guide* for more information on DLOs, saving while active, and application recovery.

SAVACTWAIT

Specifies the amount of time to wait for an object that is in use before continuing the save. If a lock is not obtained in the specified time, the object is not saved.

120: The system waits up to 120 seconds for an object lock before continuing the save operation.

***NOMAX:** No maximum wait time exists.

wait-time: Specify the time (in seconds) to wait for an object lock before continuing the save operation. Valid values range from 0 through 99,999.

Examples

Example 1: Performing a Complete Save Operation

```
SAVDLO  DLO(*ALL)  FLR(*ANY)  DEV(TAP01)
```

This command saves all folders, documents, and mail to the tape device TAP01.

Example 2: Saving All Changes

```
SAVDLO  DLO(*CHG)  DEV(TAP01)
```

This command saves all documents created or changed since the last complete save operation, folders created since the last complete save operation, and all mail.

Example 3: Saving Objects Changed After a Specific Date

```
SAVDLO  DLO(*SEARCH) DEV(TAP01)  OWNER(*ALL)
  REFCHGDATE('01/01/92')
```

This command saves all documents changed or created after 01/01/92. This command is useful for saving changes between backups of the documents. This command is similar to the Save Changed Objects (SAVCHGOBJ) used for other object types.

Example 4: Saving Documents and Folders Changed After a Specific Date

```
SAVDLO  DLO(*SEARCH) DEV(TAP01) SRCHTYPE(*ALL)
  OWNER(*ALL)  REFCHGDATE('01/01/92')
```

This command saves all folders created since 01/01/92 and all documents created or changed since 01/01/92.

Example 5: Saving Documents Created After a Specific Date

```
SAVDLO  DLO(*SEARCH) DEV(TAP01)
  CRTDATE((*AVAIL '01/01/92'))  OWNER(*ALL)
```

This command saves all documents created or changed since 01/01/92.

Example 6: Saving Documents and Folders Created After a Specific Date

```
SAVDLO  DLO(*SEARCH) DEV(TAP01) SRCHTYPE(*ALL)
  CRTDATE((*AVAIL '01/01/92'))  OWNER(*ALL)
```

This command saves all documents and folders created since 01/01/92.

Example 7: Freeing System Storage During the Save Operation

```
SAVDLO  DLO(DOCX)  FLR(FOLDERA) DEV(DKT01) STG(*FREE)
```

This command saves the document named DOCX in folder FOLDERA to the diskette device DKT01. As part of the save operation, the system storage that was occupied by the data portion of the document is freed.

Example 8: Saving Folders

```
SAVDLO  DLO(*ALL) DEV(*SAVF)  FLR(BILL TOM MARY)
  SAVF(SAVF1)
```

This command saves the folders BILL, TOM, and MARY, and all the documents in these folders.

Example 9: Saving Just the Folder Level of a Folder

```
SAVDLO  DLO(*FLRLVL)  FLR(STATUS/DEC01)
  DEV(*SAVF)  SAVF(MYLIB/SAVF1)
```

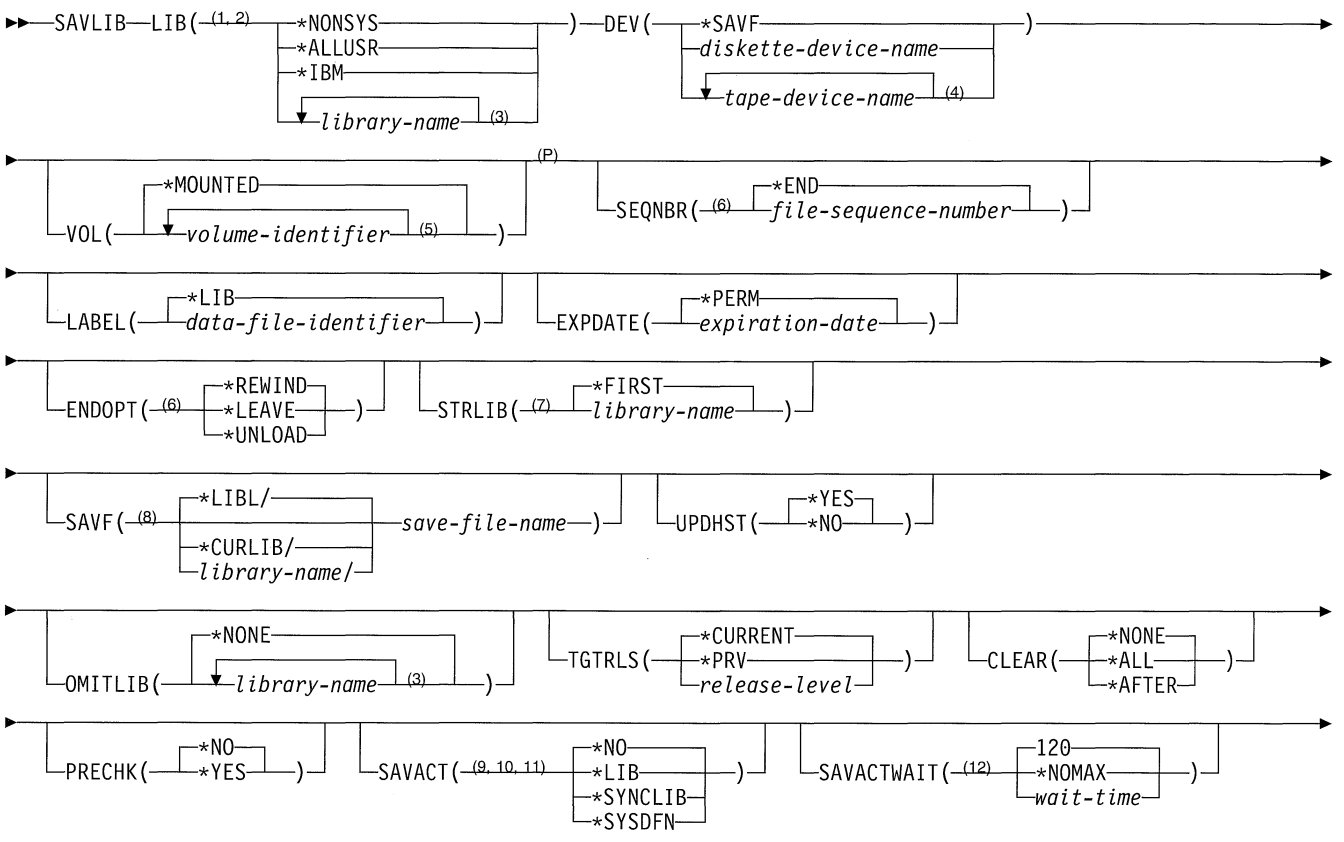
This command saves folder DEC01 in library STATUS and all the documents in this folder to save file SAVF1 in library MYLIB. Subfolders in this folder are not saved.

Example 10: Saving Just Mail

```
SAVDLO  DLO(*MAIL)  DEV(*SAVF)
```

This command saves all distribution objects and all documents referred to by a mail log.

SAVLIB (Save Library) Command



Notes:

1 Only one library can be specified when SAVF is specified

2 LIB(*NONSYS), LIB(*ALLUSR), or LIB(*IBM) is not valid when TGTRLS(*PRV) is specified

3 A maximum of 300 repetitions

4 A maximum of 4 repetitions

5 A maximum of 75 repetitions

6 Applies to tape devices only

7 STRLIB is valid only when LIB(*NONSYS), LIB(*ALLUSR), or LIB(*IBM) is specified

8 The SAVF parameter is required when DEV(*SAVF) is specified

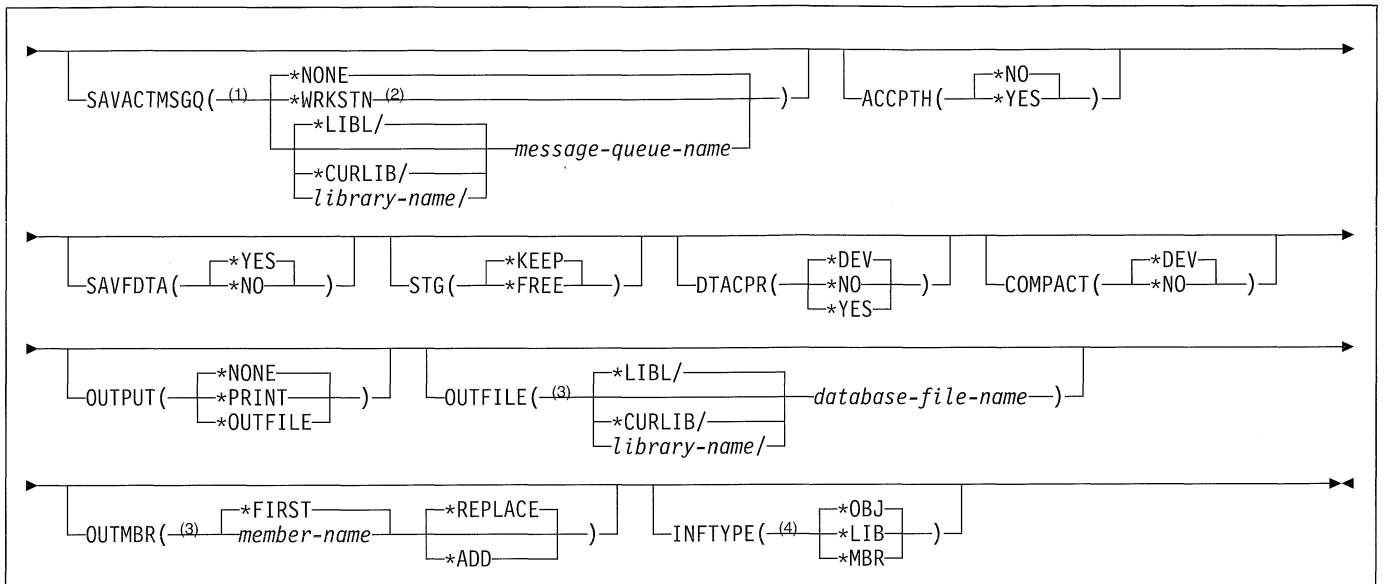
9 SAVACT(*SYNCLIB) is not valid when LIB(*IBM) or LIB(*ALLUSR) is specified

10 SAVACT(*LIB), SAVACT(*SYNCLIB), or SAVACT(*SYSDFN) is not valid when TGTRLS(V2R2M0), or TGTRLS(*PRV) pointing to V2R2M0, is specified

11 SAVACT(*LIB), SAVACT(*SYNCLIB), or SAVACT(*SYSDFN) is valid only when STG(*KEEP) is specified

12 This parameter is valid only when SAVACT(*LIB), SAVACT(*SYNCLIB), or SAVACT(*SYSDFN) is specified

P All parameters preceding this point can be specified in positional form.



Notes:

1 This parameter is valid only when SAVACT(*LIB), SAVACT(*SYNCLIB), or SAVACT(*SYSDFN) is specified.

2 This option is only valid for interactive jobs.

3 This parameter is valid only if OUTPUT(*OUTFILE) is specified.

4 This parameter is valid only if OUTPUT(*PRINT) or OUTPUT(*OUTFILE) is specified.

Purpose

The Save Library (SAVLIB) command allows the user to save a copy of a one or more libraries.

When saving to a save file, only one library can be specified.

Documents and folders contained in QDOC library can be saved by the Save Document Library Object (SAVDLO) command.

The SAVLIB command saves the entire library; this includes the library description, the object descriptions, and the contents of the objects in the library. For job queues, message queues, output queues, data queues, and logical files, only the object definitions are saved, not the contents. Logical file access paths may be saved, however, by using the ACCPTH parameter. The contents of a save file can be saved by using the Save Save File Data (SAVSAVFDTA) command or by using the SAVFDTA parameter on the SAVLIB command. The libraries and their objects are not affected in the system unless the command specifies that the storage is to be freed. However, unless UPDHST(*NO) is specified, when saving to a save file, the description of each library and each object is updated with the date, place, and time it was last saved. If a group of libraries is saved by specifying *NONSYS, *ALLUSR, or *IBM for the LIB parameter, the date, time, and place are updated in the history information for a data area in QSYS (data area QSAVLIBALL, QSAVALLUSR, or QSAVIBM).

The types of objects saved by this command are the same as those listed for the OBJTYPE parameter in "Appendix A, Expanded Parameter Descriptions," with the addition of *DTADCT. Certain OS/400 system objects that are not contained in user libraries (such as user profiles) are not saved by this command. They can be saved by the Save System (SAVSYS) or Save Security Data (SAVSECDTA) commands.

Note: This command ignores all file overrides currently in effect for the job, except for the listing file.

Restrictions:

1. To use this command, the user must have either the special authority *SAVSYS specified in the user profile by the SPCAUT parameter, or the user must have:
   - Read authority for, or be the owner of, each library specified and
   - Object existence authority for each object in the library.

   If the user does not have the correct authorities for all of the libraries and objects specified, only those for which the user does have authority are saved.
2. When saving to tape or diskette, the user must have use authority to the device description.
3. When saving to a save file, the user must have add authority and use authority to the save file.
4. All diskettes used to save the libraries should be initialized in the save/restore format. If tape is used, a standard labeled tape volume should be used.
5. No library being saved, or the objects in it, can be updated by a job that is running at the time the save operation occurs unless save-while-active is used.
6. When the contents of a save file are being saved to the same save file by specifying SAVFDTA(*YES), only the description of the save file is saved.

7. When the contents of a save file are saved by specifying *YES on the SAVFDTA parameter, the save file must be restored before objects contained in it can be restored.

Required Parameters

LIB

Specifies which libraries are saved to diskette or tape or to a save file.

Notes:

1. If the user specifies *ALLUSR or *IBM on this parameter, this command should be run when the specified libraries are not being used. If objects in a library are in use while the library is being saved, the objects are not saved. To ensure a complete save of all libraries, run this command with the system in a restricted state. For example, if SAVLIB LIB(*ALLUSR) is run when the subsystem QSNADS is active, the QAO* files are not saved in library QUSRSYS. To save the *QAO files, end the QSNADS subsystem before running SAVLIB LIB(*ALLUSR).
2. Doing a SAVLIB LIB(*IBM) and then doing a SAVLIB LIB(*ALLUSR) saves the same libraries as a SAVLIB LIB(*NONSYS), but requires two restore commands.

***NONSYS:** All user-created libraries, the QGPL library, and licensed program libraries such as QRPG and QIDU are saved. All subsystems must be ended by the End Subsystem (ENDSBS) or End System (ENDSYS) command before this option is specified. When *NONSYS is specified, the libraries are saved in alphabetical order on the media.

***ALLUSR:** All user libraries are saved. All libraries with names that do not begin with the letter Q are saved except for the following:

```
#CGULIB     #DSULIB     #SEULIB
#COBLIB     #RPGLIB
#DFULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered "user libraries", and are also saved:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

***IBM:** Saves all system (IBM) libraries except for the following Q libraries:

```
QDOC        QPFRDATA    QSPL        QTEMP
QDSNX       QRCL        QSRV        QUSER38
QGPL        QRECOVERY   QSYS        QUSRSYS
QGPL38      QRPLOBJ     QS36F       QUSRVxRxMx
```

Note: A different library name, in the format QUSRVxRxMx, may be created by the user for each previous release supported by IBM to contain any user commands to be compiled in a CL program for the previous release. For the QUSRVxRxMx user library, VxRxMx is the version, release, and modification level of a previous release that IBM continues to support.

The following libraries with names that do not begin with the letter Q are also saved:

```
#CGULIB     #DSULIB     #SEULIB
#COBLIB     #RPGLIB
#DFULIB     #SDALIB
```

library-name: Specify the names of the libraries to be saved. A maximum of 300 libraries can be specified. The names QSYS, QSRV, QTEMP, QSPL, QDOC, QRPLOBJ, and QRECOVERY cannot be specified.

Notes:

1. When several library names are specified, the overall save time is reduced, compared to that required to save objects from each of the libraries by using individual commands.
2. Only one library can be specified when saving to a save file.

DEV

Specifies the name of the device on which the libraries are saved. The device name must already be known on the system by a device description.

***SAVF:** The save operation is done using the save file specified by the Save File (SAVF) parameter.

diskette-device-name: Specify the name of the diskette device used to save the objects.

tape-device-name: Specify the names of one or more tape devices used for the save operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. Using more than one tape device permits one tape volume to be rewound and unloaded while another tape device processes the next tape volume.

Optional Parameters

VOL

Specifies the volume identifiers of the volumes on which the data is saved. The volumes must be placed in the device in the order specified on this parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The data is saved on the volumes placed in the device.

volume-identifier: Specify the identifiers of up to 75 volumes in the order they are placed in the device and used to save the system data.

SEQNBR

Specifies, only when a tape is used, which sequence number is used as the starting point for the save operation.

***END:** The save operation begins after the last sequence number on the tape volume.

file-sequence-number: Specify the sequence number of the file used as a starting point for the save operation.

If *NONSYS, *ALLUSR, or *IBM is specified on the LIB parameter, the save operation for the set of libraries begins at the sequence number specified. The first file saved in this set is the QFILE file. The QFILE file contains the list of libraries saved.

LABEL

Specifies the name that identifies the data file on the tape or diskette that is used for the save operation. If the LABEL parameter is used on the save command, the user must specify this label on the restore command.

***LIB:** The file label is created by the system using name of the library specified on the LIB parameter.

data-file-identifier: Specify the data file identifier (up to 17 characters) of the data file used for the save operation. This option is valid only for saving a single library. Do not specify *SAVLIB; the system will return an error message if *SAVLIB is specified.

EXPDATE

Specifies the expiration date. The files cannot be overwritten until the expiration date. The expiration date must be later than or equal to the current date.

***PERM:** The files are permanently protected.

expiration-date: Specify the date when protection for the file ends.

Note: For operations to diskette, the end date specified must be later than the date of the save operation. Otherwise, the ended save/restore files may be lost when the next save/restore file is written during the save operation.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

STRLIB

Specifies the library with which to begin the *NONSYS, *IBM, or *ALLUSR save.

If an irrecoverable media error occurs during the save operation, this parameter can be used to restart the operation.

Note: In the steps that follow, *NONSYS is specified on the LIB parameter. If you are restoring IBM-supplied libraries or all user-created libraries and IBM-supplied libraries, specify *IBM or *ALLUSR instead.

The basic steps for restarting a save operation are:

1. Check the job log to determine the library where the previous save operation failed. Find the last library saved, which is indicated by a successful save completion message.
2. Load the next tape and ensure the tape is initialized.
3. Issue the following command:

```
SAVLIB LIB(*NONSYS) DEV(TAP01) ENDOPT(*LEAVE)
     STRLIB(library-name) ACCPTH(*YES)
     OMITLIB(library-name)
```

   where the *library-name* for the STRLIB and OMITLIB parameters is the last library successfully saved. This starts the save operation on the library after the last successfully saved library. Specify the value for the ACCPTH parameter that was specified on the previous SAVLIB command.

   Note: Restoring the system using this set of media requires two RSTLIB SAVLIB(*NONSYS) commands to restore the libraries.

***FIRST:** The save operation begins with the first library in alphabetic order.

library-name: Specify the name of the library with which to begin the save.

SAVF

Specifies the qualified name of the save file used to contain the save data. The save file must be empty or CLEAR(*ALL) must be specified.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the save file.

UPDHST

Specifies whether the save history information of each saved object is updated with the date, time, and location of the current save operation. The save history information for an object is displayed using the Display Object Description (DSPOBJD) command. The save history information is used to determine which journal entries are processed when RCVRNG(*LASTSAVE) and FROMENT(*LASTSAVE) are used on the Apply Journaled Changes (APYJRNCHG) command.

Note: This parameter only applies when saving to a save file (DEV parameter).

***YES:** The last save date, time, and location information is updated for each object saved. This value works the same way as a save operation to diskette or tape.

***NO:** The save history information contained in the description of each object saved is not updated.

Note: UPDHST(*NO) is used only for a save operation that is not intended for recovery or saving. For example, if the save data is transmitted, record by record, to another system and the save file is immediately deleted, the user will probably not want to update the save history information.

OMITLIB

Specifies the libraries to be excluded from the save operation. This parameter is valid only if LIB(*NONSYS), LIB(*IBM), or LIB(*ALLUSR) is specified.

***NONE:** No libraries are excluded from the save operation.

library-name: Specify the name of the library to be excluded from the save operation. Up to 300 library names can be specified.

TGTRLS

Specifies the release level of the operating system on which you intend to restore and use the object.

The release level is specified in the format VxRxMx, where Vx is the version, Rx is the release, and Mx is the modification level. The release level V2R3M0 is Version 2, Release 3, Modification 0.

To specify that an object be saved for distribution to a system at a different release level than the system on which the save operation is to occur, the procedure differs for program and non-program objects and by the release level on which program objects are created. If, for example, you are saving an object for distribution to a target system running on an earlier release, you have the following choices:

For program objects

- If the program was created at a release level more current than the targeted earlier release, you must (1) create the program again specifying the targeted earlier release, (2) save the program specifying the targeted earlier release, and then (3) restore the program on the target system.
- If the program was created at the same release level as the target system, you can (1) save the program specifying the targeted earlier release and then (2) restore the program on the target system.

For non-program objects

You can (1) save the object specifying the targeted earlier release and then (2) restore the object on the target system.

Note: Not all objects can be targeted to another release. To find out which objects are supported, see the chart in the *Basic Backup and Recovery Guide*.

***CURRENT:** The object is to be restored on the release of the operating system currently running on your system. If V2R3M0 is running on the system, *CURRENT means that you intend to restore the object on a system with V2R3M0 installed. You also can restore the object on a system with any later release of the operating system installed.

***PRV:** The object is to be restored on the previous release with modification level 0 of the operating system. If V2R3M0 is running on the system, *PRV means that you intend to restore the object on a system with V2R2M0 installed. You also can restore the object on a system with any later release of the operating system installed.

release-level: Specify the release level in the format VxRxMx. The object can be restored on a system with the specified release or with any later release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. Valid values for release V2R3M0 are: V2R3M0 and V2R2M0.

You can use the following table to find out what values can be specified and what release is specified when you choose the *CURRENT and *PRV values on your operating system.

Table 72. Valid Values for TGTRLS Parameter in VxRxMx Format

Your OS/400 System Release	*CURRENT Value	*PRV Value	*Release-Level* Variables
V2R3M0	V2R3M0	V2R2M0	V2R3M0 V2R2M0
V2R2M0	V2R2M0	V2R1M0	V2R2M0 V2R1M1 V2R1M0
V2R1M1	V2R1M1	V1R3M0	V2R1M1 V2R1M0 V1R3M0
V2R1M0	V2R1M0	V1R3M0	N/A
V1R3M0	V1R3M0	V1R2M0	N/A
V1R2M0	V1R2M0	V1R1M0 V1R1M2	N/A

CLEAR

Specifies whether tapes, diskettes, or save files that contain active data and are encountered during the save operation are automatically cleared. An uncleared tape or diskette is one containing a file with an expiration date later than the date of the save operation (including files

protected permanently with EXPDATE(*PERM)). This parameter does not initialize diskettes or assign standard labels to tapes; diskettes must already be initialized in the save/restore format and tapes must be standard labeled.

Note: If a diskette is not initialized, is incorrectly named, or a tape volume that is not initialized is encountered during the save operation, an inquiry message allows the volume to be initialized or renamed.

***NONE:** None of the uncleared tapes, diskettes, or save files encountered during the save operation are automatically cleared. If the save operation cannot proceed because an uncleared tape or diskette is encountered, an inquiry message is sent to the operator, allowing the ending of the save operation, or specifying that the currently selected tape or diskette be cleared so the operation can continue. If a save file is not cleared, the inquiry message is sent to the work station message queue for an interactive job, or to the operator for a batch job. The save file must be empty, or all tapes or diskettes used to perform the save operation should be cleared before the save command is issued.

***ALL:** All uncleared tapes, diskettes, or save files encountered during the save operation are automatically cleared.

If tapes are used and a sequence number is specified the tape is cleared and, starting with that sequence number, all tapes following the first tape are cleared.

***AFTER:** All the uncleared tapes or diskettes after the first tape or diskette are automatically cleared. If the operation cannot proceed because the first tape or diskette is uncleared, an inquiry message is sent to the system operator, allowing ending of the operation or specifying that the currently selected tape be cleared so the operation can continue.

PRECHK

Specifies whether the save operation for a library should end if all objects in the library specified by this command do not satisfy the following conditions of the save operation:

- The objects exist,
- They are not found to be damaged,
- They are not locked by another job, and
- The requester of the save operation has authority to save the objects.

Note: When multiple libraries are specified, PRECHK(*YES) applies to each library independently. For example, the failure to save one library does not affect the saving of other libraries. However, if PRECHK(*YES) and SAVACT(*SYNCLIB) are specified and an object in any library to be saved does not pass the preliminary check, the save operation ends and no libraries are saved.

***NO:** Allows the save operation for a library to continue, saving only objects that can be saved.

***YES:** Ends the save operation for a library before any data is written if, after all objects in the specified library are checked, one or more objects cannot be saved. If multiple libraries are specified, the save operation continues with the next library.

SAVACT

Specifies whether an object can be updated while it is being saved.

Note: If your system is in a restricted state and the SAVACT parameter is specified, the save operation is performed as if SAVACT(*NO) was specified.

***NO:** Objects that are in use are not saved. Objects cannot be updated while being saved.

***LIB:** Objects in a library can be saved while they are in use by another job. All of the objects in a library reach a checkpoint together and are saved in a consistent state in relationship to each other.

Note: Libraries with thousands of objects may be too large for this option.

***SYNCLIB:** Objects in a library can be saved while they are in use by another job. All of the objects and all of the libraries in the save operation reach a checkpoint together and are saved in a consistent state in relationship to each other.

***SYSDFN:** Objects in a library can be saved while they are in use by another job. Objects in a library may reach checkpoints at different times and may not be in a consistent state in relationship to each other.

Note: Specifying this value eliminates some size restrictions and may enable a library to be saved that could not be saved with SAVACT(*LIB).

SAVACTWAIT

Specifies the amount of time to wait for a commit boundary or an object that is in use before continuing the save. If a lock is not obtained in the specified time, the object is not saved. If a commit boundary is not reached in the specified time, the save operation is ended.

120: The system waits up to 120 seconds for a commit boundary or an object lock before continuing the save operation.

***NOMAX:** No maximum wait time exists.

wait-time: Specify the time (in seconds) to wait for a commit boundary or an object lock before continuing the save operation. Valid values range from 0 through 99,999.

SAVACTMSGQ

Specifies the message queue that the save operation uses to notify the user that the checkpoint processing for a library is complete. A separate message is sent for

each library to be saved when SAVACT(*SYSDFN) or SAVACT(*LIB) is specified. When SAVACT(*SYNCLIB) is specified, one message is sent for all libraries in the save operation.

This parameter can be used to save the objects at a known, consistent boundary to avoid additional recovery procedures following a restore operation. Applications can be stopped until the checkpoint processing complete message is received.

***NONE:** No notification message is sent.

***WRKSTN:** The notification message is sent to the work station message queue.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue.

ACCPTH

Specifies whether the logical file access paths that are dependent on the physical files being saved are also saved. The access paths are saved only if all members on which the access paths are built are included in this save operation. Informational messages are sent indicating the number of logical file access paths saved with each physical file. All physical files on which an access path is built must be in the same library. This parameter does not save logical file objects; it controls only the saving of the access paths. More information on the restoring of saved access paths is in the *Basic Backup and Recovery Guide*.

Attention!

If the based-on physical files and the logical files are in different libraries, the access paths are saved.

However, if the logical files and the based-on physical files are in different libraries and the logical files or physical files do not exist at restore time (such as during disaster recovery or the files were deleted) the access paths are not restored. They are rebuilt.

For the fastest possible restore operation for logical files, the logical files and the based-on physical files must be in the same library and must be saved at the same time.

***NO:** Only objects in the libraries specified on the command are saved. No logical file access paths are saved.

***YES:** The physical files in the specified libraries and all eligible logical file access paths over them are saved.

SAVFDTA

Specifies, for save file objects, whether the description of a save file, or both the description and the contents of a save file, are saved on a tape, diskette, or in another save file.

***YES:** The description and contents of the save file are saved.

***NO:** Only the description of a save file is saved.

STG

Specifies whether the system storage occupied by the data portion of files, modules, programs, service programs, Structured Query Language (SQL) packages, and journal receivers in the library being saved is freed as part of the save operation. Only the data portion of the objects is freed, not the descriptions of the objects.

***KEEP:** The storage occupied by the data portion of the objects being saved is not freed.

***FREE:** The storage occupied by the data portion of the files (except for save files), modules, programs, service programs, SQL packages, and journal receivers being saved is freed as part of the saved operation. The storage for all the objects in a library is freed only after all the objects in that library are saved successfully.

Note: To prevent the possible abnormal end of a program, the program being saved must not be running in the system if *FREE is specified.

DTACPR

Specifies whether data compression is performed.

***DEV:** If the tape device has the hardware compression feature installed, processing proceeds as if DTACPR(*YES) is specified. If the compression feature is not installed or if save data is written to a diskette or save file, processing proceeds as if DTACPR(*NO) is specified.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device data compression are performed if supported on the device.

***NO:** No data compression is done.

***YES:** If the save operation is to tape and the target device has the hardware compression feature, hardware compression is done. If the feature is not present, or if the save data is written to diskette or save file, software data compression is done. If the save operation is being done while other jobs on the system are active and soft-

ware data compression is used, the overall system performance may be affected.

COMPACT
Specifies whether device data compaction is performed.

***DEV:** Device data compaction is performed if the data is saved to tape and all tape devices specified on the DEV parameter support the compaction feature.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device data compression are performed if supported on the device.

***NO:** Device data compaction is not performed.

OUTPUT
Specifies whether a list of information about the saved libraries is created. The information can be printed with the job's spooled output or directed to a database file.

***NONE:** No output listing is created.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: If OUTPUT(*OUTFILE) is specified, you must specify the database file name on the OUTFILE parameter.

OUTFILE
Specifies the qualified name of the database file to which the information about the object is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QASAVOBJ in QSYS with the format name QSRSAV as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which the output of the command is directed.

OUTMBR
Specifies the name of the database file member to which the output is directed when OUTPUT(*OUTFILE) is specified.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that receives the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it. If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

INFTYPE
Specifies the type of information which is printed or directed to the database file.

***OBJ:** The list contains an entry for each object requested to be saved.

***LIB:** The list contains library entry for each library requested to be saved.

***MBR:** The list contains an entry for each object or, for database files, each member requested to be saved.

Examples

Example 1: Saving a Library on a Tape Device

```
SAVLIB  LIB(JOE)  DEV(TAP01)
```

This command saves the library named JOE on the tape that is in the tape device named TAP01. The storage occupied by JOE in the system is not freed.

Example 2: Saving on Multiple Volumes

```
SAVLIB  LIB(QGPL)  DEV(DKT01) VOL(ABC DEF GHI)
```

The general purpose library (QGPL) is saved on the diskettes in the device named DKT01. The diskettes used must have the volume names ABC, DEF, and GHI. If the save operation is not finished when diskette ABC is full, a message is issued to the operator asking for volume DEF to be placed in the device.

Example 3: Freeing Storage when Saving Data

```
SAVLIB   LIB(CUSDATA) DEV(DKT01) VOL(CUSNM CUSAD)
  STG(*FREE)
```

The library named CUSTDATA is saved on volumes CUSNM and CUSAD, which are put in the diskette device DKT01.

The storage occupied by the files, modules, programs, service programs, SQL packages, and journal receivers in the CUSTDATA library is freed after it is saved.

Example 4: Saving on Multiple Devices

```
SAVLIB  LIB(QGPL)  DEV(TAP01 TAP02 TAP03)
 VOL(USRA USRB USRC USRD) ENDOPT(*UNLOAD)
```

The library named USRLIB is saved on four tape volumes on three tape devices. The volume named USRA is put on the device named TAP01, the volume named USRB on the device named TAP02, the volume named USRC on the device named TAP03, and the volume named USRD on the device named TAP01. The volume named USRA is rewound, and must be unloaded by the operator when processing is complete so that the device named TAP01 can be used for the volume named USRD.

Example 5: Saving a Library with a Media File Label

```
SAVLIB  LIB(LIB1)  DEV(TAP01) LABEL(MONDAYBACKUP)
```

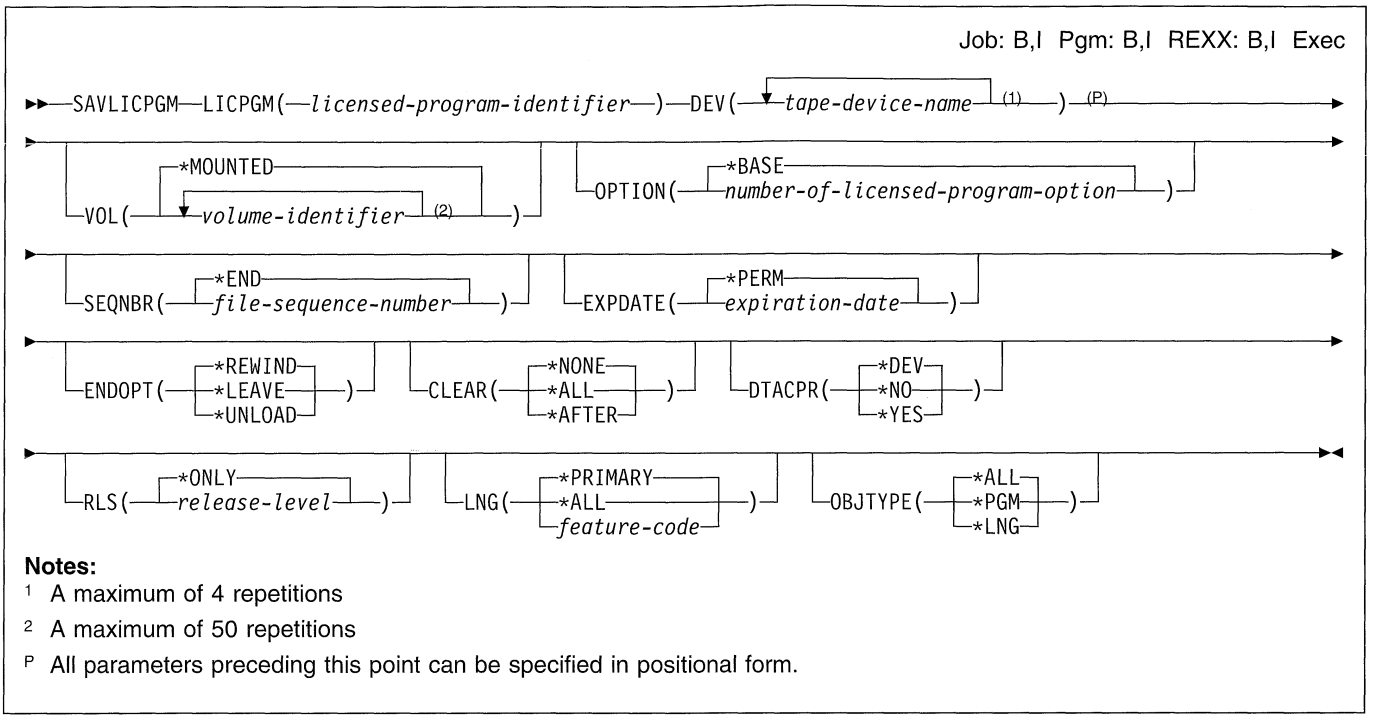
This command uses the tape device named TAP01 to save the library named LIB1 on tape. The library is saved with a media file label of MONDAYBACKUP. This label must be specified when restoring the library or any of its objects.

Example 6: Specifying Where the Save Operation Starts

```
SAVLIB  LIB(*NONSYS)  DEV(TAP01 TAP02) STRLIB(MIKESLIB)
```

This command saves the nonsystem libraries beginning with the library named MIKESLIB on tape devices named TAP01 and TAP02. The nonsystem libraries are saved in alphabetic order. Therefore, all libraries beginning with and following the name MIKESLIB are saved.

SAVLICPGM (Save Licensed Program) Command



Notes:
1 A maximum of 4 repetitions
2 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Save Licensed Program (SAVLICPGM) command saves a copy of the objects that make up a licensed program. It saves the licensed program in a form that can be restored by the Restore Licensed Program (RSTLICPGM) command.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. A tape volume with a standard label must be placed in the tape device.
3. Some licensed programs can only be saved if the user is enrolled in the system distribution directory. See the publication for each licensed program for a description of this restriction.
4. This command does not save code or language objects for the base OS/400 system.

Required Parameters

LICPGM

Specifies the 7-character identifier of the licensed program to be saved. The licensed programs shipped by IBM are listed in the *Licensed Programs and New Release Installation Guide*.

DEV

Specifies the names of the tape devices used for the save licensed program operation. Each device name must be known on the system by a device description. If multiple devices are specified, they must have compatible media formats. If more than one tape device is used, specify the names of the devices in the order in which they are used. Up to four device names can be specified. When more than one tape volume is used, user may want to use more than one tape device so that one tape volume can rewind/unload while another tape device is processing. Use the Work with Device Descriptions (WRKDEVD) command to display the names of the tape devices available on this system.

Optional Parameters

VOL

Specifies the volume identifiers of the volumes on which the data is saved. The volumes must be placed in the device in the order specified on this parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The licensed program data is saved on whatever volumes are placed in the tape device.

volume-identifier: Specify the identifiers of one or more volumes in the order they are placed in the tape device and used to save the licensed program.

OPTION

Specifies which of the optional parts of the licensed program given in the LICPGM parameter are saved.

***BASE:** Only the base part of the licensed program is saved.

number-of-licensed-program-option: Specify the number of the optional part of the listed licensed program that is saved.

SEQNBR

Specifies which sequence number is used for the save operation.

***END:** The system saves the file to the sequence number after the last sequence number on the tape.

file-sequence-number: Specify the sequence number of the file used for the save operation.

EXPDATE

Specifies the expiration date. The files cannot be overwritten until the expiration date. The expiration date must be later than or equal to the current date.

***PERM:** The save/restore files are protected permanently.

expiration-date: Specify the date when the save/restore files are no longer protected.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is rewound, but not unloaded, when the save operation is completed.

***LEAVE:** The tape should be left in its current position when the operation is completed; it is not rewound or unloaded.

***UNLOAD:** The tape is rewound and unloaded when the save operation is completed.

CLEAR

Specifies whether tapes, diskettes, or save files that contain active data and are encountered during the save operation are automatically cleared. An uncleared tape or diskette is one containing a file with an expiration date later than the date of the save operation (including files protected permanently with EXPDATE(*PERM)). This parameter does not initialize diskettes or assign standard labels to tapes; diskettes must already be initialized in the save/restore format and tapes must be standard labeled.

***NONE:** None of the uncleared tapes that are encountered during the save operation are automatically cleared. If the save operation cannot proceed because an uncleared tape is encountered, an inquiry message is sent to the operator, allowing the operator to end the save operation, or to specify that the currently selected tape be cleared so the operation can continue. All tapes used to perform the save operation should be cleared before the save command is issued.

***ALL:** All uncleared tapes encountered during the save operation are automatically cleared. If a sequence number is specified, the tape is cleared and, starting with that sequence number, all tapes following the first tape are cleared.

***AFTER:** All the uncleared tapes after the first tape are automatically cleared. If the operation cannot proceed because the first tape is uncleared, an inquiry message is sent to the system operator, allowing the operator to end the operation or to specify that the currently selected tape be cleared so the operation can continue.

DTACPR

Specifies whether data compression is performed.

***DEV:** If the tape device has the hardware compression feature installed, processing proceeds as if DTACPR(*YES) is specified. If the compression feature is not present, or if save data is written to diskette or a save file, processing proceeds as if DTACPR(*NO) is specified.

***NO:** No data compression is performed.

***YES:** Program data compression is performed. If the program is saved to tape and the target device has the hardware compression feature, hardware compression is performed. If the feature is not present, software compression is performed. If the save operation is running while other jobs on the system are active, and software compression is used, there may be an effect on the overall system performance.

RLS

Specifies which version, release, and modification level of the product is saved.

***ONLY:** Only one version, release, and modification level is installed for the product option.

release-level: Specify the release level in the format VxRxMy, where Vx is the version number, Rx is the release number, and My is the modification number. Valid values for *x* are the numbers 0 through 9. Valid values for *y* are the numbers 0 through 9 and the letters A through Z.

LNG

Specifies which National Language Version (NLV) is used for the save operation.

Note: This parameter is ignored when OBJTYPE(*PGM) is specified.

***PRIMARY:** The primary language is saved. The primary language is the language of the operating system.

Note: Use the GO LICPGM function with option 20 to display the primary language of the operating system.

***ALL:** All languages are saved.

feature-code: Specify the NLV identifier for the language that is saved for the licensed program. The IBM-supplied language feature codes are listed in the

Licensed Programs and New Release Installation Guide or can be displayed using GO LICPGM, option 20.

OBJTYPE

Specifies the type of licensed program objects being saved.

***ALL:** Program and language objects specified on the LNG parameter are saved.

***PGM:** Only the program objects for the licensed program are saved.

***LNG:** The objects associated with the NLV identified on the LNG parameter are saved.

Examples

Example 1: Tapes Cleared Automatically

```
SAVLICPGM  LICPGM(5738WP1)  DEV(TAP01)  CLEAR(*ALL)
```

This command saves the *BASE option of the OfficeVision/400 licensed program. It is saved on the tape that is in the TAP01 tape drive. Each uncleared tape is cleared automatically when it is encountered, and the save operation continues without operator intervention.

Example 2: Saving on Labeled Tape Volume

```
SAVLICPGM  LICPGM(5738WP1)  DEV(TAP01)  VOL(ABCDE)
```

The *BASE option of the OfficeVision/400 licensed program is saved on the TAP01 tape drive, starting on the tape volume labeled ABCDE. If the save operation exceeds the storage capacity of one tape, a message requesting that another volume be placed in the TAP01 tape drive is shown to the operator.

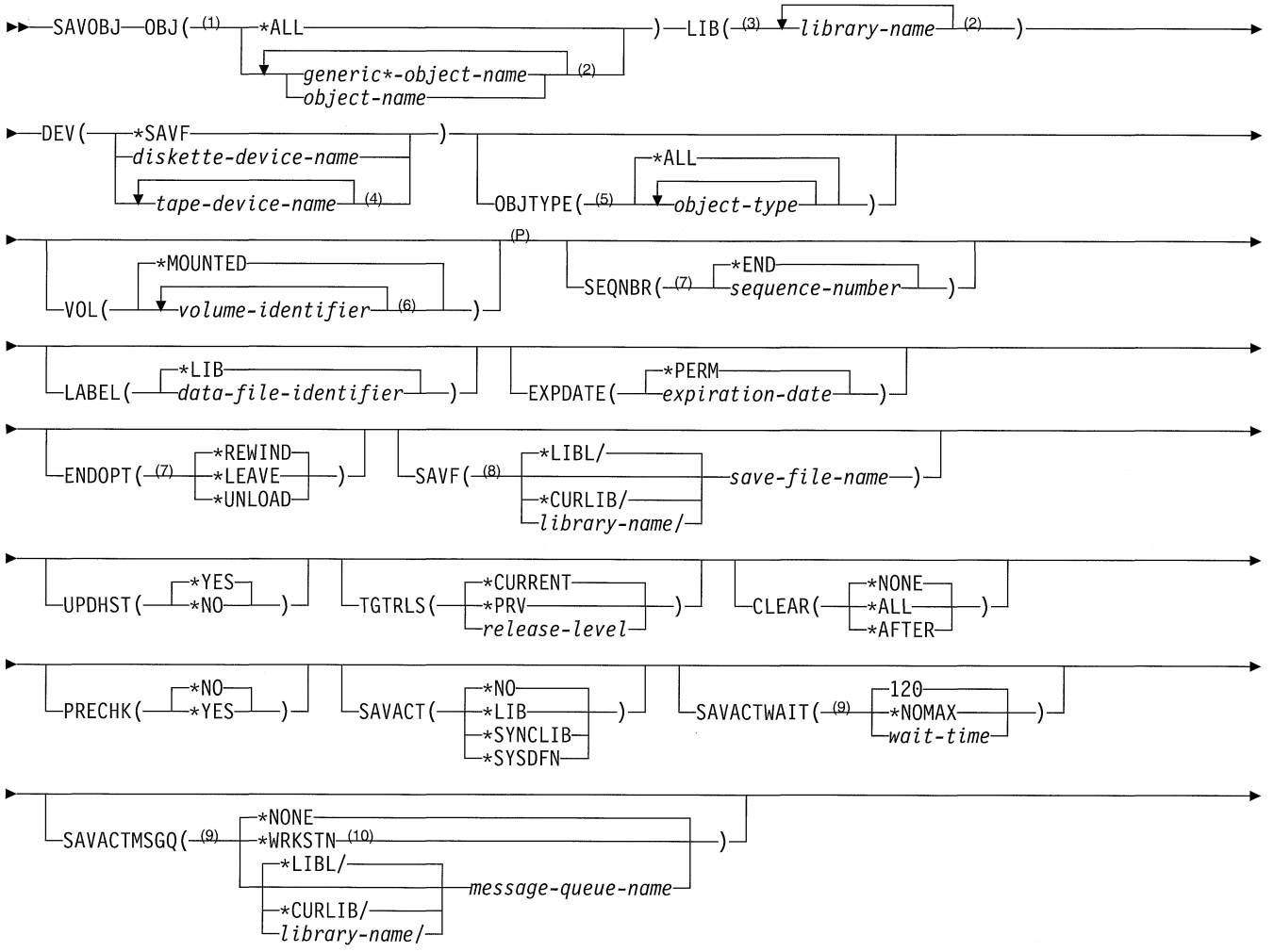
Example 3: Saving on Multiple Volumes

```
SAVLICPGM  LICPGM(5738SS1)  OPTION(2)  DEV(TAP01 TAP02)
```

The online help option of the OS/400 system is saved on tape drives TAP01 and TAP02 in alternating order. If the save operation exceeds the storage capacity of two tapes, a message requesting that another volume be put on TAP01 is shown to the operator. The tapes are rewound at the completion of the save operation.

SAVOBJ (Save Object) Command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

1 OBJ(*ALL) is required when more than one library is specified

2 A maximum of 300 repetitions

3 Only one library can be specified when SAVF is used.

4 A maximum of 4 repetitions

5 A list of the valid OS/400 object types for this command is in Appendix A.

6 A maximum of 75 repetitions

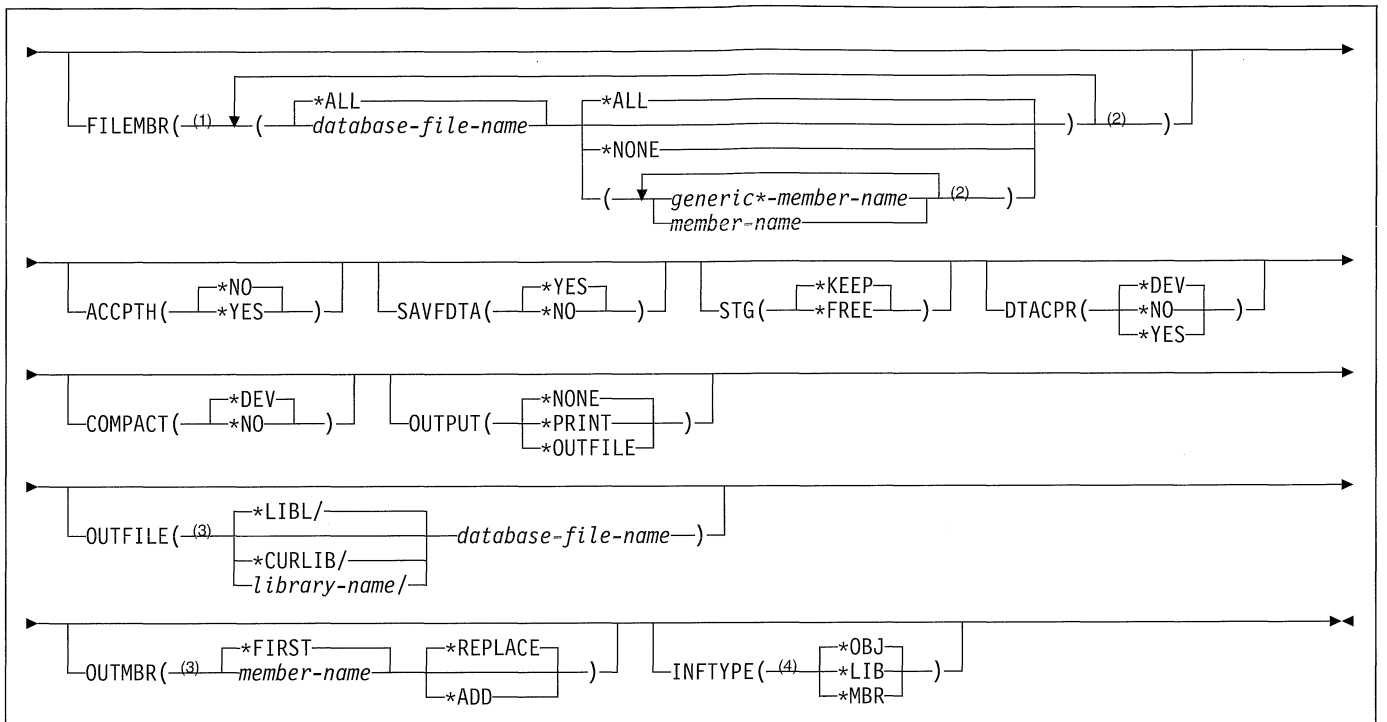
7 Applies to tape devices only

8 The SAVF parameter is required when DEV(*SAVF) is specified

9 This parameter is valid only when SAVACT(*LIB), SAVACT(*SYNCLIB), or SAVACT(*SYSDFN) is specified

10 This option is only valid for interactive jobs.

P All parameters preceding this point can be specified in positional form.



Notes:

1 This parameter cannot be specified if STG(*FREE) is specified.

2 A maximum of 50 repetitions

3 This parameter is valid only if OUTPUT(*OUTFILE) is specified.

4 This parameter is valid only if OUTPUT(*OUTFILE) or OUTPUT(*PRINT) is specified.

Purpose

The Save Object (SAVOBJ) command saves a copy of a single object or a group of objects located in the same library. When OBJ(*ALL) is specified, objects can be saved from up to 300 libraries. When saving to a save file, only one library can be specified. The types of objects that can be saved by this command are listed in the OBJTYPE parameter. The system saves the specified objects by writing a copy of each one on diskettes, tapes, or in a save file. Objects in the system are not affected unless the command specifies that the storage should be freed. However, the description of each object is changed with the date, time, and place when it was last saved unless UPDHST(*NO) is specified when saving to a save file.

For job queues, output queues, data queues, message queues, and logical files, only the object descriptions are saved, and the contents of the objects are not saved. However, logical file access paths can be saved by the use of the ACCPTH parameter. The contents of a save file can be saved by use of the Save Save File Data (SAVSAVFDTA) command or by specifying the SAVFDTA parameter on this command.

Note: This command ignores all file overrides currently in effect for the job, except for the output file.

Restrictions:

1. To use this command, the user must have either the special authority *SAVSYS specified in the user profile by the SPCAUT parameter, or have (a) object existence authority for each object specified, and (b) read authority for the specified library. If the user does not have the necessary authority to a specified object, all objects except that one are saved.
2. SAVOBJ does not save the data dictionary for the library or its associated database files. To save them, the SAVLIB command should be used.
3. When saving to tape or diskette, the user must have use authority for the device description and device file.
4. When saving to a save file, the user must have add authority and use authority for the save file.
5. All diskettes used to save the objects must be initialized in the save/restore format.
6. If tape is used, a standard-labeled volume must be placed in the device.
7. No object being saved can be changed by a job that is running at the time the save operation occurs unless save-while-active is used.
8. When the contents of a save file are being saved to the same save file by specifying SAVFDTA(*YES), only the description of the save file is saved.
9. When the contents of a save file are saved by specifying *YES on the SAVFDTA parameter, the save file must be restored before objects contained in it can be restored.

Required Parameters

OBJ

Specifies the names of one or more objects, or the generic name of each group of objects, to save. The objects being saved must be in the library specified in the LIB parameter.

If the OBJTYPE parameter is not specified, all the object types listed in the description of the OBJTYPE parameter are saved, provided they are in the specified library and have the specified names.

***ALL:** Objects in the specified libraries are saved, depending on the values specified for the OBJTYPE parameter.

generic-object-name:* Specify the generic name of the object. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk (*) substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix, for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

object-name: Specify one or more names of specific objects to save. Both generic names and specific names can be specified in the same command.

LIB

Specifies which libraries contain the objects to be saved.

DEV

Specifies the name of the device on which the objects are saved. The device name must already be known on the system by a device description.

***SAVF:** The save operation is done by using the save file specified by the save file (SAVF) parameter.

diskette-device-name: Specify the name of the diskette device used to save the objects.

tape-device-name: Specify the names of one or more tape devices used for the save operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. Using more than one tape device permits one tape volume to be rewound and unloaded while another tape device processes the next tape volume.

Optional Parameters

OBJTYPE

Specifies the types of the OS/400 system objects to save. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ALL:** All object types specified by name and in the specified library are saved. If *ALL is also specified on the OBJ parameter, all the objects in the library that are of types that can be saved are saved.

object-type: Specify the value for each of the types of objects to be saved.

The object types saved are also the ones saved and restored by the Save Changed Object (SAVCHGOBJ), Save Library (SAVLIB), Restore Object (RSTOBJ), and the Restore Library (RSTLIB) commands. Data dictionaries and the associated files are saved only by using the SAVLIB command.

VOL

Specifies the volume identifiers of the volumes on which the data is saved. The volumes must be placed in the device in the order specified on this parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The data is saved on the volumes placed in the device.

volume-identifier: Specify the identifiers of one or more volumes in the order in which they are placed in the device and are used to save the objects. A maximum of 75 volume identifiers can be specified.

If more volumes are needed than are specified, the data is saved on the additional volumes placed in the device.

SEQNBR

Specifies, only when tape is used, which sequence number to use for saving the objects.

***END:** The system saves the objects after the last sequence number on the tape.

sequence-number: Specify the sequence number of the file.

LABEL

Specifies the name that identifies the data file on the tape or diskette that is used for the save. If the LABEL parameter is used on the save command, the user must specify this label on the restore command.

***LIB:** The file label is created by the system using name of the library specified on the LIB parameter.

data-file-identifier: Specify the data file identifier (up to 17 characters) of the data file used for the save operation. This option is valid only for saving a single library.

EXPDATE

Specifies the expiration date. The files cannot be overwritten until the expiration date. The expiration date must be later than or equal to the current date.

Note: This parameter is valid for tape and diskette.

***PERM:** The file is protected permanently.

expiration-date: Specify the when protection for the file ends.

Note: For save operations to diskette, the expiration date specified must be later than the date of the save operation. Otherwise, the save and restore files whose expiration date has been exceeded

may be lost when the next save and restore file is written during the save operation.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

Note: If no objects are saved (for example, no objects meet the selection value specified on the command), the tape volume is not opened and the ENDOPT parameter is ignored.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

SAVF

Specifies the qualified name of the save file used to contain the save data. The save file must be empty or CLEAR(*ALL) must be specified.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the save file.

UPDHST

Specifies whether the save history information of each saved object is changed with the date, time, and location of the save operation. The save history information for an object is displayed using the Display Object Description (DSPOBJD) command. The save history information is used to determine which journal entries are processed when RCVRNG(*LASTSAVE) and FROMENT(*LASTSAVE) are used on the Apply Journaled Changes (APYJRNCHG) command.

Note: This parameter only applies when saving to a save file (*SAVF on the DEV parameter).

***YES:** The last save date, time, and location information is changed for each object saved. This value works the same way as a save operation to diskette or tape.

***NO:** The save history information contained in the description of each object saved are not changed.

Note: UPDHST(*NO) should only be used for a save operation that is not intended for recovery or save. For example, if the save data is sent, record by record, to another system and the save file immediately deleted, the save history information is probably not to be updated.

TGTRLS

Specifies the release level of the operating system on which you intend to restore and use the object.

The release level is specified in the format VxRxMx, where Vx is the version, Rx is the release, and Mx is the modification level. The release level V2R3M0 is Version 2, Release 3, Modification 0.

To specify that an object be saved for distribution to a system at a different release level than the system on which the save operation is to occur, the procedure differs for program and non-program objects and by the release level on which program objects are created. If, for example, you are saving an object for distribution to a target system running on an earlier release, you have the following choices:

For program objects

- If the program was created at a release level more current than the targeted earlier release, you must (1) create the program again specifying the targeted earlier release, (2) save the program specifying the targeted earlier release, and then (3) restore the program on the target system.
- If the program was created at the same release level as the target system, you can (1) save the program specifying the targeted earlier release and then (2) restore the program on the target system.

For non-program objects

You can (1) save the object specifying the targeted earlier release and then (2) restore the object on the target system.

Note: Not all objects can be targeted to another release. To find out which objects are supported, see the chart in the *Basic Backup and Recovery Guide*.

***CURRENT:** The object is to be restored on the release of the operating system currently running on your system. If V2R3M0 is running on the system, *CURRENT means that you intend to restore the object on a system with V2R3M0 installed. You also can restore the object on a system with any later release of the operating system installed.

***PRV:** The object is to be restored on the previous release with modification level 0 of the operating system. If V2R3M0 is running on the system, *PRV means that you intend to restore the object on a system with V2R2M0 installed. You also can restore the object on a

system with any later release of the operating system installed.

release-level: Specify the release level in the format VxRxMx. The object can be restored on a system with the specified release or with any later release of the operating system installed.

Valid values depend on the current version, release, and modification level, and they change with each new release. Valid values for release V2R3M0 are: V2R3M0 and V2R2M0.

You can use the following table to find out what values can be specified and what release is specified when you choose the *CURRENT and *PRV values on your operating system.

Table 73. Valid Values for TGTRLS Parameter in VxRxMx Format

Your OS/400 System Release	*CURRENT Value	*PRV Value	*Release-Level* Variables
V2R3M0	V2R3M0	V2R2M0	V2R3M0 V2R2M0
V2R2M0	V2R2M0	V2R1M0	V2R2M0 V2R1M1 V2R1M0
V2R1M1	V2R1M1	V1R3M0	V2R1M1 V2R1M0 V1R3M0
V2R1M0	V2R1M0	V1R3M0	N/A
V1R3M0	V1R3M0	V1R2M0	N/A
V1R2M0	V1R2M0	V1R1M0 V1R1M2	N/A

CLEAR

Specifies whether tapes, diskettes, or save files that contain active data and are encountered during the save operation are automatically cleared. An uncleared tape or diskette is one containing a file with an expiration date later than the date of the save operation (including files protected permanently with EXPDATE(*PERM)). This parameter does not initialize diskettes or assign standard labels to tapes; diskettes must already be initialized in the save/restore format and tapes must be standard labeled.

Note: If a diskette that is not initialized is incorrectly named, or a tape volume that is not initialized is encountered during the save operation, an inquiry message allows the volume to be initialized or renamed.

***NONE:** None of the uncleared tapes, diskettes, or save files encountered during the save operation are automatically cleared. If the save operation cannot proceed because an uncleared tape or diskette is encountered, an inquiry message is sent to the operator, allowing the ending of the save operation, or specifying that the currently selected tape or diskette be cleared so the operation can continue. If a save file is not cleared, the inquiry message is sent to the work station message queue for an interactive job, or to the operator for a batch job. The save file must be empty, or all tapes or diskettes used to perform the save operation should be cleared before the save command is issued.

***ALL:** All uncleared tapes, diskettes, or save files encountered during the save operation are automatically cleared.

If tapes are used and a sequence number is specified the tape is cleared and, starting with that sequence number, all tapes following the first tape are cleared.

***AFTER:** All the uncleared tapes or diskettes after the first tape or diskette are automatically cleared. If the operation cannot proceed because the first tape or diskette is uncleared, an inquiry message is sent to the system operator, allowing ending of the operation or specifying that the currently selected tape be cleared so the operation can continue.

PRECHK

Specifies whether the save operation for a library ends if all objects specified by this command do not satisfy the following conditions:

- The objects exist
- They are not found to be damaged
- They are not locked by another job
- The requester of the save operation has authority to save the objects

***NO:** The save operation for a library continues, saving only objects that can be saved.

***YES:** The save operation for a library ends before any data is written if, after specified objects are checked, one or more objects cannot be saved. If multiple libraries are specified, the save operation continues with the next library. However, if PRECHK(*YES) and SAVACT(*SYNCLIB) are specified and an object in any library to be saved does not meet the preliminary check conditions, the save operation ends and no objects are saved.

SAVACT

Specifies whether an object can be updated while it is being saved.

Note: If your system is in a restricted state and the SAVACT parameter is specified, the save operation is performed as if SAVACT(*NO) was specified.

***NO:** Objects that are in use are not saved. Objects cannot be updated while being saved.

***LIB:** Objects in a library can be saved while they are in use by another job. All of the objects in a library reach a checkpoint together and are saved in a consistent state in relationship to each other.

Note: Libraries with thousands of objects may be too large for this option.

***SYNCLIB:** Objects in a library can be saved while they are in use by another job. All of the objects and all of

the libraries in the save operation reach a checkpoint together and are saved in a consistent state in relationship to each other.

***SYSDFN:** Objects in a library can be saved while they are in use by another job. Objects in a library may reach checkpoints at different times and may not be in a consistent state in relationship to each other.

Note: Specifying this value eliminates some size restrictions and may enable a library to be saved that could not be saved with SAVACT(*LIB).

SAVACTWAIT

Specifies the amount of time to wait for a commit boundary or an object that is in use before continuing the save. If a lock is not obtained in the specified time, the object is not saved. If a commit boundary is not reached in the specified time, the save operation is ended.

120: The system waits up to 120 seconds for a commit boundary or an object lock before continuing the save operation.

***NOMAX:** No maximum wait time exists.

wait-time: Specify the time (in seconds) to wait for a commit boundary or an object lock before continuing the save operation. Valid values range from 0 through 99,999.

SAVACTMSGQ

Specifies the message queue that the save operation uses to notify the user that the checkpoint processing for a library is complete. A separate message is sent for each library to be saved when SAVACT(*SYSDFN) or SAVACT(*LIB) is specified. When SAVACT(*SYNCLIB) is specified, one message is sent for all libraries in the save operation.

This parameter can be used to save the objects at a known, consistent boundary to avoid additional recovery procedures following a restore operation. Applications can be stopped until the checkpoint processing complete message is received.

***NONE:** No notification message is sent.

***WRKSTN:** The notification message is sent to the work station message queue.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue.

FILEMBR

Specifies the database file members to be saved. This parameter cannot be specified if STG(*FREE) is specified.

This parameter has two parts: the file name and the member name.

Element 1: Database File Names

***ALL:** The member list following this value applies to all files indicated by the OBJ parameter.

database-file-name: Specify the name of the database file that contains the specified members to be saved. Up to 50 of the file/member list combinations can be specified for a single command.

Restrictions:

- Each database file specified on the FILEMBR parameter must also be specified on the OBJ parameter by either its complete name, a generic name, or *ALL.
- The OBJTYPE parameter either must be *ALL, or it must include *FILE.
- Generic names are not valid for the database file name, but are allowed for the member name.
- Duplicate file names are not allowed.

Element 2: Member Names

***ALL:** All members are saved from the specified file.

***NONE:** No members are saved from the specified file.

generic-member-name:* Specify the generic name of the member. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk (*) substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix, for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

member-name: Specify the names of the members saved from the specified file.

Restrictions:

- If nongeneric member names are specified, the specified members must exist in the file for any part of the file to be saved or restored.
- If generic member names are specified, the file must contain member names that match the generic names for the file to be saved. For example, if PAY* is specified as a generic member name, and the system is unable to find a member whose name starts with PAY, the file is not saved. If files specified by the FILEMBR parameter are not saved because members with the specified generic name cannot be found, a diagnostic message is sent, the save operation ends, and an escape message is sent specifying the number of files not saved. If at

least one of the files processed for the FILEMBR parameter contains a member with the specified generic name, the diagnostic message is not sent, and the number of files not saved is in the final completion message.

ACCPTH

Specifies whether the logical file access paths that are dependent on the physical files being saved are also saved. The access paths are saved only if all members on which the access paths are built are included in this save operation. Informational messages are sent indicating the number of logical file access paths saved with each physical file. All physical files on which an access path is built must be in the same library. This parameter does not save logical file objects; it controls only the saving of the access paths. More information on the restoring of saved access paths is in the *Basic Backup and Recovery Guide*.

Attention!

If the based-on physical files and the logical files are in different libraries, the access paths are saved.

However, if the logical files and the based-on physical files are in different libraries and the logical files or physical files do not exist at restore time (such as during disaster recovery or the files were deleted) the access paths are not restored. They are rebuilt.

For the fastest possible restore operation for logical files, the logical files and the based-on physical files must be in the same library and must be saved at the same time.

***NO:** Only objects specified on the command are saved. No logical file access paths are saved.

***YES:** The specified physical files and all eligible logical file access paths over them are saved.

SAVFDTA

Specifies, for save file objects, whether the description of a save file, or both the description and the contents of a save file, are saved on a tape, diskette, or in another save file.

***YES:** The description and contents of the save file are saved. If SAVFDTA(*YES) and STG(*FREE) are specified, storage is freed for the save file.

***NO:** Only the description of a save file is saved.

STG

Specifies whether the system storage occupied by the data portion of the files, modules, programs, service programs, Structured Query Language (SQL) packages, and journal receivers being saved is freed after the save operation is finished. Only the data portion of the object is freed, not the object's description.

***KEEP:** The storage occupied by the data portion of the objects being saved is not freed.

***FREE:** The storage occupied by the data portion of the files (except for save files), modules, programs, service programs, SQL packages, and journal receivers is freed as part of the save operation. The storage is freed only after all objects are saved successfully.

Note: To prevent the possible abnormal end of a program, the program being saved must not be running in the system if *FREE is specified.

DTACPR

Specifies whether data compression is performed.

***DEV:** If the tape device has the hardware compression feature installed, processing proceeds as if DTACPR(*YES) is specified. If the compression feature is not installed or if save data is written to a diskette or save file, processing proceeds as if DTACPR(*NO) is specified.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device data compression are performed if supported on the device.

***NO:** No data compression or decompression occurs.

***YES:** If the save operation is to tape and the target device has the hardware compression feature, hardware compression is done. If the feature is not present or if the save data is written to a diskette or save file, software compression is done. If the save operation is running while other jobs on the system are active and software compression is used, overall system performance may be affected.

COMPACT

Specifies whether device data compaction is performed.

***DEV:** Device data compaction is performed if the data is saved to tape and all tape devices specified on the DEV parameter support the compaction feature.

***NO:** Device data compaction is not performed.

OUTPUT

Specifies whether a list of information about the saved objects and members is created. The information can be printed with the job's spooled output or directed to a database file.

***NONE:** No output listing is created.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: If OUTPUT(*OUTFILE) is specified, you must specify the database file name on the OUTFILE parameter.

OUTFILE

Specifies the qualified name of the database file to which the information about the object is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QASAVOBJ in QSYS with the format name QSRSAV as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which the output of the command is directed.

OUTMBR

Specifies the name of the database file member to which the output is directed when OUTPUT(*OUTFILE) is specified.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that receives the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it. If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

INFTYPE

Specifies the type of information which is printed or directed to the database file.

***OBJ:** The list contains an entry for each object requested to be saved.

***LIB:** The list contains an entry for each library requested to be saved.

***MBR:** The list contains an entry for each object or, for database files, each member requested to be saved.

Examples

Example 1: Saving Program and File With Identical Names

```
SAVOBJ  OBJ(PETE)  LIB(LIBX)  DEV(DKT01)
```

This command saves the objects named PETE which are located in the LIBX library. If, for example, LIBX contains both a program and a file named PETE, both objects are saved. The storage occupied by the object is not freed because the STG parameter default (*KEEP) was assumed.

Example 2: Freeing System Storage

```
SAVOBJ  OBJ(MSTRPAY PAY*)  LIB(QGPL)  DEV(DKT01)
  STG(*FREE)
```

The object named MSTRPAY, and all the objects whose names start with the characters PAY located in the general purpose library (QGPL), are saved. The objects are copied on diskettes. As part of the save operation, the system storage that was occupied by the data portion of the saved file, module, program, service program, SQL package, and journal receiver objects is freed.

Example 3: Saving File on Diskette

```
SAVOBJ   OBJ(FILEA)  OBJTYPE(*FILE)  LIB(LIBY)
  DEV(DKT01)  VOL(TOM)  CLEAR(*ALL)
```

The file named FILEA in the LIBY library is saved on the diskette. The diskette must be identified by the volume identifier TOM. If the diskette was not cleared, it is cleared automatically before FILEA is saved on it.

Example 4: Saving Objects Supported on Previous Release

```
SAVOBJ  OBJ(PAY*)  LIB(LIB1)  DEV(TAP01)
  TGTRLS(*PRV)
```

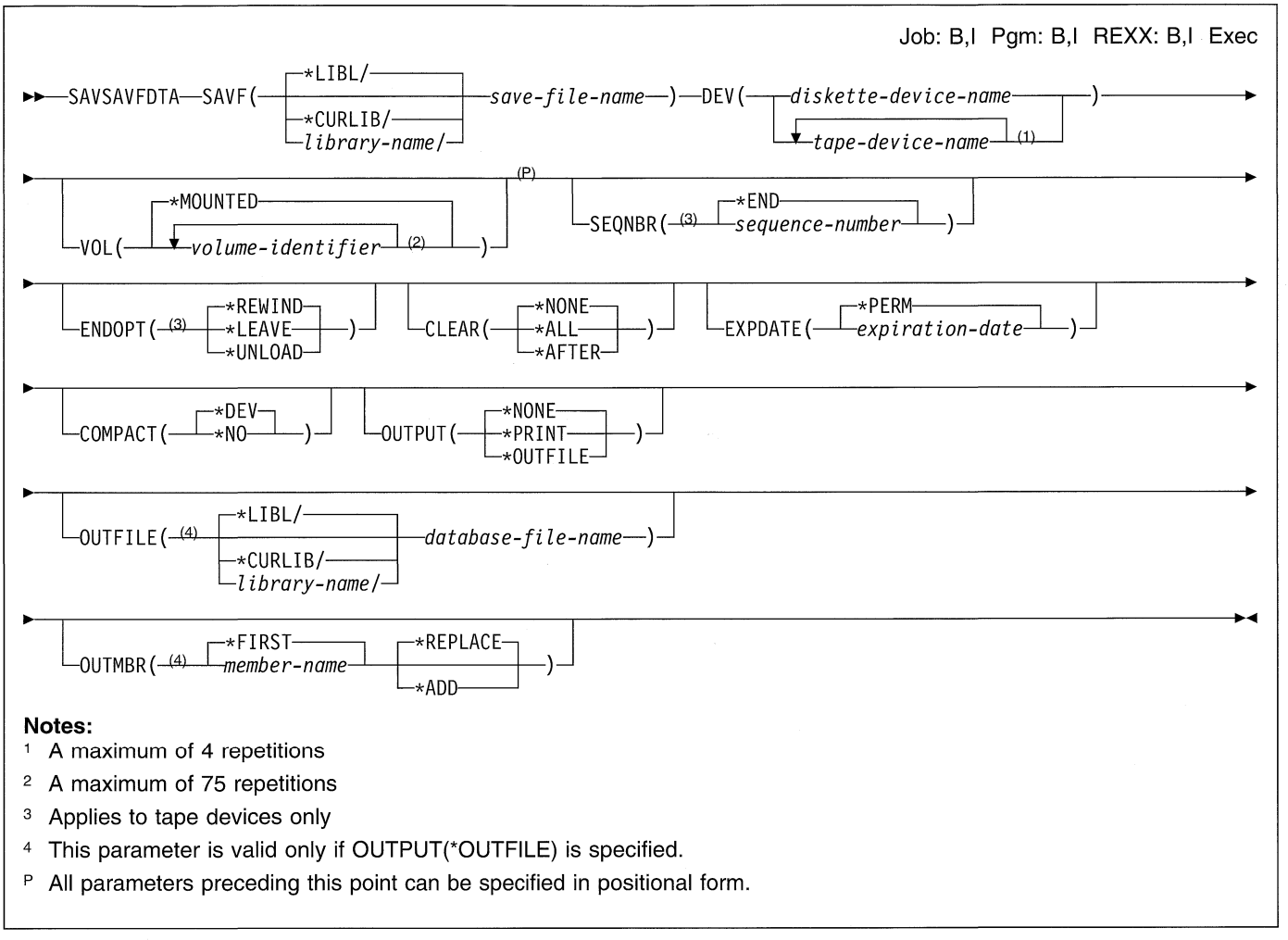
This command saves the objects beginning with the characters PAY from the LIB1 library in a format compatible with the previous release of the OS/400 system. Only those objects supported on the previous release are saved.

Example 5: Saving Description and Data of File

```
SAVOBJ  OBJ(SAVEFILE)  LIB(MYLIB)  OBJTYPE(*FILE)
  DEV(DKT01)  SAVFDTA(*YES)
```

This command saves the file named SAVEFILE which is located in the library named MYLIB. Both the description and the data are saved for this save file.

SAVSAVFDTA (Save Save File Data) Command



Notes:

1 A maximum of 4 repetitions

2 A maximum of 75 repetitions

3 Applies to tape devices only

4 This parameter is valid only if OUTPUT(*OUTFILE) is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Save Save File Data (SAVSAVFDTA) command saves the contents of a save file to a tape or diskette. This command saves the save data in the save file to the device in a way that allows the user to restore objects directly from diskette or tape.

A save file containing data created by the Save Security Data (SAVSECDTA) or Save Configuration (SAVCFG) command can only be saved to tape.

The information written on tape or diskette by this command is similar to the data that was previously written to the save file by the save command that originally created the save file data. This includes the object descriptions, and object contents that existed when the original save operation was done.

This command uses only the save file and device description objects; it does not refer to or modify the description or contents of the objects included in the file save data. Thus, objects included in the save file are not locked during the running of this command, and the save history information (date, place, and time when each object was last saved) is not updated by this command for each object in the save file.

The description of the save file is not included in the save operation (unless it was included with the objects that were saved to create the save data in the file). In addition, this command does not update the save history information for the save file object, so the last save operation date, time, and place always identify the last save operation of the save file object description, not its contents.

Note: This command ignores all file overrides currently in effect for the job, except for the output file.

Restrictions:

1. The user of this command must have use authority for the save file and use authority for the tape or diskette device description.
2. All diskettes to be used to perform the save operation must be initialized in the save/restore format. If a tape is used, it must have a standard label. Also, the save file cannot be in use by a job running at the time the save operation occurs.

Required Parameters

SAVF

Specifies the qualified name of the save file whose contents are saved. If no library qualifier is specified, *LIBL is used to find the file.

Note: The save file must contain the data made by the run of a previous save command or an error message is sent, and its contents are not saved to diskette or tape.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the save file.

DEV

Specifies the names of the devices used for the save operation. Each device name must already be known on the system by a device description.

diskette-device-name: Specify the name of the diskette device used to save the data.

tape-device-name: Specify the name of one or more tape devices that are used for the save operation. If more than one tape device is used, specify the names of the devices in the order they are used. Up to four device names can be specified. When more than one tape volume is used, it may be advantageous to use more than one tape device so that tape volumes can be rewound and/or unloaded while another tape device is being processed.

Optional Parameters

VOL

Specifies the volume identifiers of the volumes on which the data is saved. The volumes must be placed in the device in the order specified on this parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The data is saved on the volumes placed in the device.

volume-identifier: Specify the identifiers of one or more volumes in the order they are placed in the device and used to perform the save operation.

SEQNBR

Specifies, when tape is used, the sequence number on the tape where the save operation begins.

***END:** The system saves the data after the last sequence number on the tape.

sequence-number: Specify the sequence number at which the save operation begins. The sequence number specified must either exist on the tape, or be one greater than the sequence number of the last file on the tape. Specifying a sequence number of 1 always causes the save operation to start at the beginning of the tape.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

Note: This parameter is ignored if a diskette device is specified on the DEV parameter.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

CLEAR

Specifies whether uncleared tapes or diskettes encountered during the save operation are automatically cleared. An uncleared tape or diskette is one containing a file with an expiration date later than the date of the save operation, which includes files protected permanently using EXPDATE(*PERM). This parameter does not control initializing diskettes; they must already be initialized in the save/restore format, and tapes should have standard labels.

***NONE:** None of the uncleared tapes or diskettes encountered during the save operation are automatically cleared. If the save operation cannot proceed because an uncleared tape or diskette is encountered, an inquiry message is sent to the operator, allowing the operator to end the save operation, or to specify that the currently selected tape or diskette be cleared so the operation can continue. All tapes or diskettes used to perform the save operation should be cleared before the save command is issued.

***ALL:** All uncleared tapes or diskettes encountered during the save operation are automatically cleared.

If tapes are used and a sequence number is specified, the tape is cleared, starting with that sequence number, and all tapes following the first tape are cleared.

***AFTER:** All the uncleared tapes or diskettes after the initial tape or diskette are automatically cleared. If the operation cannot proceed because the first tape or diskette is uncleared, an inquiry message is sent to the system operator, allowing the operator to end the opera-

tion or to specify that the currently selected tape or diskette be cleared so the operation can continue.

EXPDATE
Specifies the expiration date. The files cannot be overwritten until the expiration date. The expiration date must be later than or equal to the current date.

PERM: The file written on diskette or tape is protected permanently.

expiration-date: Specify the date on which protection of the file on diskette or tape ends. The date must be specified in job date format.

COMPACT
Specifies whether device data compaction is performed.

***DEV:** Device data compaction is performed if the data is saved to tape and all tape devices specified on the DEV parameter support the compaction feature.

***NO:** Device data compaction is not performed.

OUTPUT
Specifies whether a list with information about the saved contents of a save file is created. The information can be printed with the job's spooled output or directed to a database file.

***NONE:** No output listing is created.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: If OUTPUT(*OUTFILE) is specified, you must specify the database file name on the OUTFILE parameter.

OUTFILE
Specifies the qualified name of the database file to which the information about the object is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QASAVOBJ in QSYS with the format name QSRSAV as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which the output of the command is directed.

OUTMBR
Specifies the name of the database file member to which the output is directed when OUTPUT(*OUTFILE) is specified.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that receives the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it. If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

Examples

Example 1: Clearing All Diskettes and Tapes Encountered

```
SAVSAVFDTA  SAVF(ONLINE)  DEV(TAP01)
  SEQNBR(1)  CLEAR(*ALL)
```

This command saves the contents of save file ONLINE to the first file on the tape volume on device TAP01. Files that have not ended on either the first tape volume or on subsequent volumes are overwritten without an inquiry message because CLEAR(*ALL) is specified.

Example 2: Restoring Library from Tape

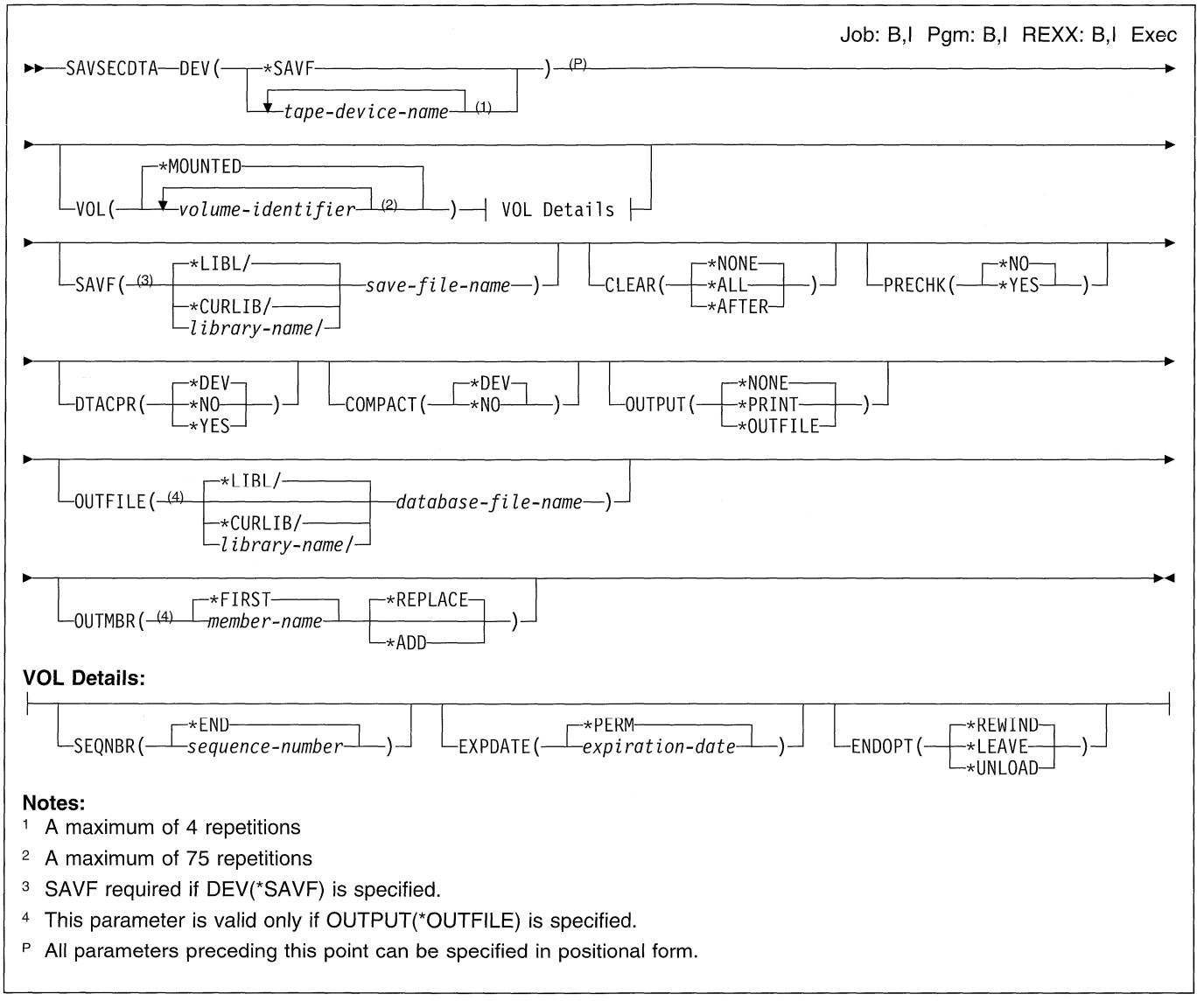
If, in the last save command to save library USRLIB, the save file was:

```
SAVLIB  USRLIB  SAVF(ONLINE)
```

Then the following command must be specified to restore the library from tape:

```
RSTLIB  USRLIB DEV(TAP01)  VOL(*MOUNTED)
```

SAVSECDTA (Save Security Data) Command



Notes:

1 A maximum of 4 repetitions

2 A maximum of 75 repetitions

3 SAVF required if DEV(*SAVF) is specified.

4 This parameter is valid only if OUTPUT(*OUTFILE) is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Save Security Data (SAVSECDTA) command saves all security information without requiring a system in a restricted state.

SAVSECDTA saves the same security information that is saved when a SAVSYS command is run including:

- User Profiles
- Authorization Lists
- Authority Holders

Information saved with the SAVSYS command or SAVSECDTA command can be restored with the RSTUSRPRF and RSTAUT commands, but a dedicated system is required.

Restrictions:

1. Users of the SAVSECDTA command must have *SAVSYS special authority.
2. Changes made to user profiles while the SAVSECDTA command is being run may not be reflected in the media, depending on when the changes occurred in relation to the save operation.
3. Concurrent running of other SAVSECDTA commands is not allowed.
4. If PRECHK(*YES) is specified and a security object cannot be saved, the save operation ends.

Required Parameter

DEV

Specifies the name of the device on which the security data is saved. The device name must already be known on the system by a device description.

***SAVF:** The save operation is done by using the save file specified by the save file (SAVF) parameter.

tape-device-name: Specify the names of one or more tape devices used for the save operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. Using more than one tape device permits one tape volume to be rewound and unloaded while another tape device processes the next tape volume.

Optional Parameters

VOL

Specifies the volume identifiers of the volumes on which the data is saved. The volumes must be placed in the device in the order specified on this parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The data is saved on the volumes placed in the device.

volume-identifier: Specify the identifiers of one or more volumes in the order in which they are placed in the device and used to save the system data. A maximum of 75 volume identifiers can be specified.

SEQNBR

Specifies, only when a tape is used, which sequence number is used as the starting point for the save operation.

***END:** The system saves the file after the last sequence number on the tape.

sequence-number: Specify the sequence number of the file.

EXPDATE

Specifies the expiration date. The files cannot be overwritten until the expiration date. The expiration date must be later than or equal to the current date.

***PERM:** The file is protected permanently.

expiration-date: Specify the date when the file protection ends.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

SAVF

Specifies the qualified name of the save file used to contain the save data. The save file must be empty or CLEAR(*ALL) must be specified.

The name of the save file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

save-file-name: Specify the name of the save file.

CLEAR

Specifies whether uncleared tapes or save files encountered during the save operation are automatically cleared. An uncleared tape is one containing a file with an expiration date later than the date of the save operation (including files protected permanently with EXPDATE(*PERM)). This parameter does not control standard-labeling tapes; tapes must be standard labeled.

Note: If an uninitialized tape volume is encountered during the save operation, a message prompts the user to initialize and rename it.

***NONE:** None of the uncleared tapes or save files encountered during the save operation are automatically cleared. If the save operation cannot proceed because an uncleared tape is encountered, an inquiry message is sent that allows the operator either to end the save operation, or to specify that the currently selected tape be cleared so the operation can continue. If a save file is not cleared, the inquiry message is sent to the work station message queue for an interactive job, or to the operator for a batch job. The save file should be empty, or all tapes used to perform the save operation should be cleared before the save command is issued.

***ALL:** Uncleared tapes or save files encountered during the save operation are automatically cleared.

If tapes are being used and a sequence number is specified, the tape starting with that sequence number and all tapes with sequence numbers that follow that of the first tape are cleared.

***AFTER:** All the uncleared tapes after the initial tape are automatically cleared. If the operation cannot proceed because the specified sequence number on the first tape volume is uncleared, an inquiry message is sent asking the system operator to end the operation or to specify that the currently selected tape be cleared so the operation can continue. This value is not valid for save operations to save files.

PRECHK

Specifies whether the save operation ends if all security objects specified do not satisfy the following conditions of the save operation:

- The objects exist
- They were not previously found to be damaged
- They are not locked by another job
- The requester of the save operation has authority to save the objects

***NO:** The save operation continues, saving only security objects that can be saved.

***YES:** The save operation ends before any data is written if, after all security objects are checked, one or more objects cannot be saved.

DTACPR

Specifies whether data compression is performed.

***DEV:** If the tape device has the hardware compression feature installed, processing proceeds as if DTACPR(*YES) is specified. If the compression feature is not installed or if save data is written to a diskette or save file, processing proceeds as if DTACPR(*NO) is specified.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device data compression are performed if supported on the device.

***NO:** No data compression occurs.

***YES:** Hardware compression is performed if the save operation is to tape and the target device has the hardware compression feature. If the feature is not present or if the save data is written to a save file, software compression is performed. Running the save operation while other jobs on the system are active and software compression is used may affect the overall system performance.

COMPACT

Specifies whether device data compaction is performed.

***DEV:** Device data compaction is performed if the data is saved to tape and all tape devices specified on the DEV parameter support the compaction feature.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device data compression are performed if supported on the device.

***NO:** Device data compaction is not performed.

MAIL

Note:

The MAIL parameter has been removed from this command for Release V2R3M0 and all subsequent releases.

OUTPUT

Specifies whether a list with information about the saved security objects is created. The information can be printed with the job's spooled output or directed to a database file.

***NONE:** No output listing is created.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: If OUTPUT(*OUTFILE) is specified, you must specify the database file name on the OUTFILE parameter.

OUTFILE

Specifies the qualified name of the database file to which the information about the object is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QASAVOBJ in QSYS with the format name QSRSAV as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which the output of the command is directed.

OUTMBR

Specifies the name of the database file member to which the output of the command is directed.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that receives the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it. If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

Examples

Example 1: Automatically Clearing Uncleared Tapes

```
SAVSECDTA  DEV(TAP01)  CLEAR(*ALL)
```

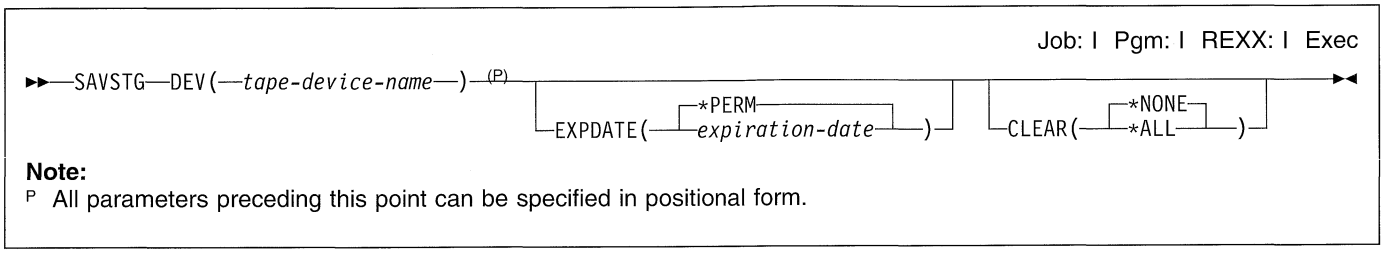
This command saves the security information, including user profiles, authorization lists, authority holders. They are saved on the TAP01 tape drive. CLEAR(*ALL) automatically clears all uncleared tapes when they are encountered.

Example 2: Sending Message When Storage Capacity Exceeded

```
SAVSECDTA  DEV(TAP01)  VOL(ABC)
```

This command saves the security information on the TAP01 tape drive, starting on the tape volume labeled ABC. If the save operation exceeds the storage capacity of one tape, a message requesting that another volume be put on the TAP01 tape drive is shown to the operator.

SAVSTG (Save Storage) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Save Storage (SAVSTG) command saves a copy of the licensed internal code and the contents of auxiliary storage (except unused space and temporary objects) to tape. This function is intended for disaster recovery backup. Individual libraries or objects cannot be restored from a save storage tape.

This command issues the PWRDWNSYS (Power Down System) command with OPTION(*IMMED) and RESTART(*YES) specified. While the system is powered down, a dedicated service tool (DST) that saves all system storage is called. At that time, a standard labeled tape volume must be placed in the tape device. Additional volumes are requested as needed. Hardware data compression is used if it is supported by the tape device. After the save system storage function is complete, an initial program load (IPL) takes place.

During the IPL after a save storage operation, a completion message is sent to the system operator message queue. The save history information for the data area QSAVSTG in library QSYS is updated with the date and time when the system storage data was saved. To show the information in this data area, use the Display Object Description (DSPOBJD) command with DETAIL(*FULL).

The restore storage operation is done using the appropriate option on the DST menu. During the IPL, after a restore storage operation, a completion message is sent to the system operator message queue, and the last restore date and time history information in the QSAVSTG data area is updated with the current date and time. In addition, the data portion of the QSAVSTG data area is updated with the date of the save storage tape used in the restore system storage operation.

Note: Because media errors cause the save operation to start over from the last tape volume, use of this command is recommended for smaller systems only.

Restrictions:

1. To use this command, you must have *SAVSYS special authority.
2. The system cannot be running any other jobs; (ENDSBS SBS (*ALL) or ENDSYS must be issued).
3. Tapes created using this command that will be used for installation should be initialized with a density that is supported by the current alternative IPL tape unit. If this is not done, the current IPL tape will have to be changed to a tape device that supports the density of the created SAVSTG tapes before installation begins.
4. Tapes created using the SAVSTG command should not be used for automatic installation.

Required Parameters

DEV

Specifies the name of the tape device used for the save storage operation. Specify the name of the tape device.

Optional Parameters

EXPDATE

Specifies the expiration date. The files cannot be overwritten until the expiration date. The expiration date must be later than or equal to the current date.

***PERM:** The tape files are permanently protected.

expiration-date: Specify the date when the tape files end. The date must be specified in the job date format. If separators, specified by the system value QDATSEP, are used, the value must be enclosed in apostrophes.

CLEAR

Specifies whether uncleared tapes encountered during the save operation are automatically cleared. An uncleared tape is one containing a file with an expiration date later than the date of the save operation (including files protected permanently with EXPDATE(*PERM)). This parameter does not control initializing tapes; tapes must be standard labeled.

Note: If a tape volume that is not initialized is encountered during the save operation, an inquiry message allows the operator to initialize the volume.

***NONE:** None of the uncleared tapes encountered during the save operation are automatically cleared. If the save operation cannot continue because an uncleared tape is encountered, an inquiry message is sent to the operator allowing the save operation to end, specifying that the currently selected tape should be cleared so the operation can continue. All tapes used to

perform the save operations should be initialized before the save command is issued.

***ALL:** All uncleared tapes encountered during the save operation are automatically cleared.

Examples

Example 1: Specifying Expiration Date

```
SAVSTG  DEV(TAP01)  EXPDATE(122290)  CLEAR(*ALL)
```

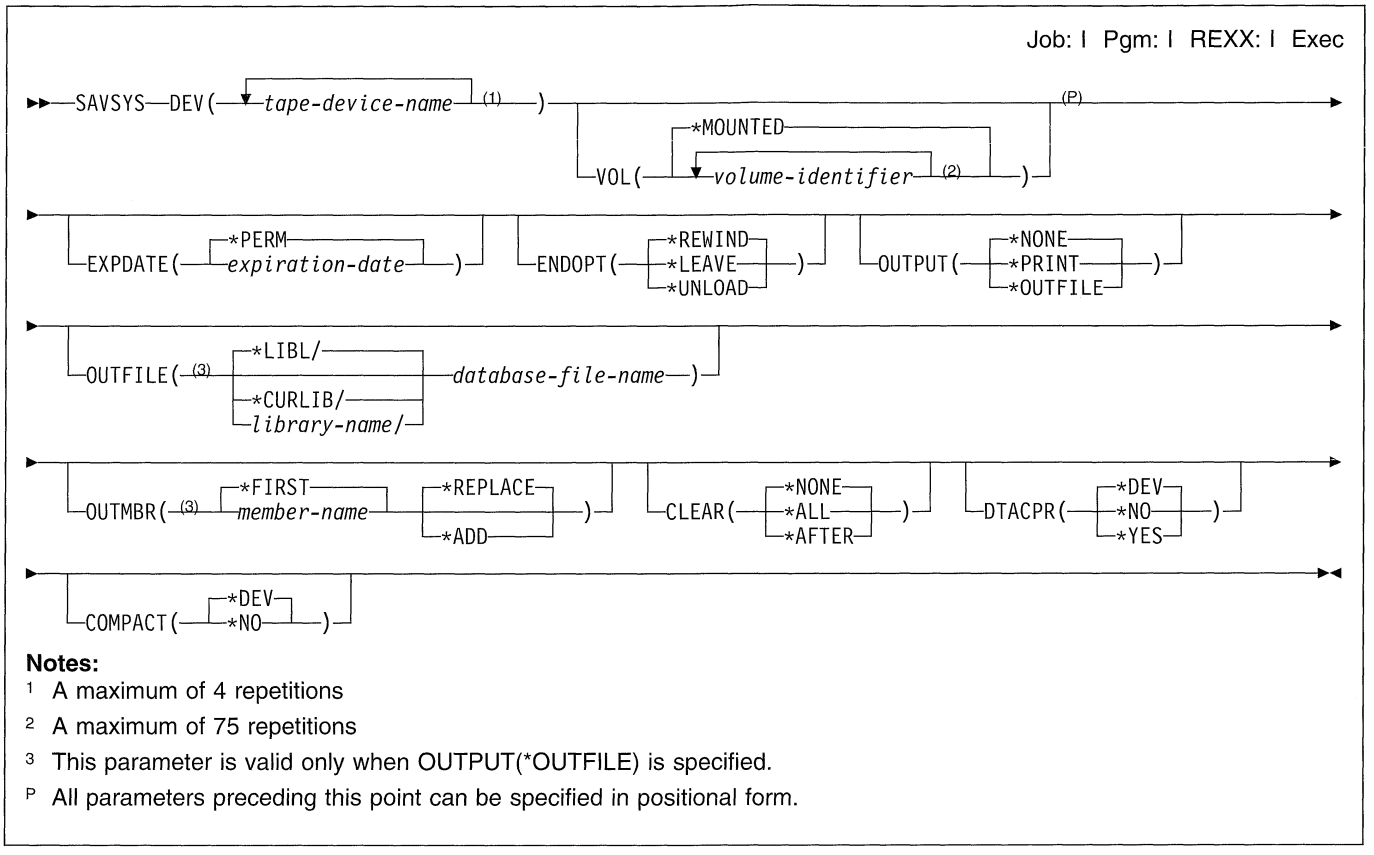
This command saves the system storage on the tape put on the TAP01 tape drive. Each uncleared tape is cleared automatically. The tape files written are protected and cannot be overwritten until December 22, 1990.

Example 2: Saving System Storage

```
SAVSTG  DEV(TAP02)
```

The system storage is saved on tape drive TAP02. CLEAR was not specified, so uncleared tapes encountered during the save operation cause an inquiry message to be sent to the operator, who either ends the save operation or specifies that the currently selected tape be cleared so the operation can continue. Because EXPDATE also was not specified, the tape files being written are protected permanently.

SAVSYS (Save System) Command



Notes:

1 A maximum of 4 repetitions

2 A maximum of 75 repetitions

3 This parameter is valid only when OUTPUT(*OUTFILE) is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Save System (SAVSYS) command saves a copy of the licensed internal code and the QSYS library in a format compatible with the installation of the AS/400 system. It does not save objects from any other library. In addition, it saves the security and configuration objects that can also be saved using the SAVSECDTA and SAVCFG commands.

To save the system data on offline storage, the system writes a copy of objects to be saved on tape. The libraries and objects are not affected in the system. This command cannot be used to free any space occupied by these objects. The history information for the data area QSAVUSRPRF in QSYS is updated with the date, time, and place where user profiles are saved. The history information for the data area QSAVSYS in QSYS is updated with the date, time, and place where the system is saved. The history information for the data area QSAVCFG in QSYS is updated with the date, time, and place where configuration objects are saved. The history information is not updated for the individual objects. To display the information in these data areas, specify the Display Object Description (DSPOBJD) command, and specify DETAIL(*FULL). Save the information from the display of QSAVUSRPRF for the location where the user profiles are saved.

When using this command, it is important to use the tape device on the system that is defined as the initial program load (IPL) device. The IPL device was defined by the service representative when the system was installed. If an IPL device is not used when using this command, then the system cannot be restored using the SAVSYS tapes (if densities or media types are incompatible).

Restrictions:

1. You must have *SAVSYS special authority to use this command.
2. All subsystems must be inactive before the SAVSYS command can be specified. The End System (ENDSYS) or End Subsystem (ENDSBS) command can be used to make the subsystems inactive. You must have job control (*JOBCTL) authority to use the ENDSYS or the ENDSBS command.
3. Tapes created using this command that will be used for installation should be initialized with a density that is supported by the current IPL tape unit. If this is not done, the current IPL tape will have to be changed to a tape device that supports the density of the created SAVSYS tapes before installation begins.
4. Tapes created using the SAVSYS command should not be used for automatic installation.

Required Parameter

DEV

Specifies the names of the tape devices used for the save system operation. Each device name must be known on the system by a device description. If multiple devices are specified, they must have compatible media formats.

If more than one tape device is used, specify the names of the devices in the order in which they are used. Up to four device names can be specified. When more than one tape volume is used, more than one tape device may be used so that one tape volume can be rewound and unloaded while another tape device is being processed.

Optional Parameters

VOL

Specifies the volume identifiers of the volumes on which the data is saved. The volumes must be placed in the device in the order specified on this parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The data is saved on the volumes placed in the device.

volume-identifier: Specify the identifiers of one or more volumes in the order they are placed in the device and used to save the system data.

EXPDATE

Specifies the expiration date. The files cannot be overwritten until the expiration date. The expiration date must be later than or equal to the current date.

***PERM:** The files are permanently protected.

expiration-date: Specify the date when protection of the files ends.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

OUTPUT

Specifies whether a list with information about the saved objects is created. The information can be printed with the job's spooled output or directed to a database file. One label entry is created for each of the files on tape.

***NONE:** No output listing is created.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

Note: If OUTPUT(*OUTFILE) is specified, you must specify the database file name on the OUTFILE parameter.

OUTFILE

Specifies the qualified name of the database file to which the information about the object is directed when *OUTFILE is specified on the OUTPUT parameter. If the file does not exist, this command creates a database file in the specified library. If a new file is created, the system uses QASAVOBJ in QSYS with the format name QSRSAV as a model.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file to which the output of the command is directed when OUTPUT(*OUTFILE) is specified.

OUTMBR

Specifies the name of the database file member to which the output of the command is directed.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the name of the file member that receives the output. If OUTMBR(*member-name*) is specified and the member does not exist, the system creates it. If the member exists, the user can add records to the end of the existing member or clear the existing member and add the records.

Element 2: Operation to Perform on Member

***REPLACE:** The existing records in the specified database file member are replaced by the new records.

***ADD:** The new records are added to the existing information in the specified database file member.

CLEAR

Specifies whether uncleared tapes encountered during the save operation are automatically cleared. An uncleared tape is one containing a file with an expiration

date later than the date of the save operation (including files protected permanently with EXPDATE(*PERM)). This parameter does not control standard-labeling tapes; tapes must be standard labeled.

Note: If a tape volume that is not initialized is encountered during the save operation, an inquiry message reminds the user to initialize the volume.

***NONE:** None of the uncleared tapes encountered during the save operation are automatically cleared. If the save operation cannot proceed because an uncleared tape is encountered, an inquiry message is sent to the operator, allowing the ending of the save operation, or specifying that the currently selected tape be cleared so the operation can continue. All tapes used to perform the save operation should be cleared before the save command is issued.

***ALL:** All uncleared tapes encountered during the save operation are automatically cleared.

***AFTER:** All the uncleared tapes after the first tape are automatically cleared. If the operation cannot proceed because the first tape is uncleared, an inquiry message is sent to the system operator, allowing him to end the operation or to specify that the currently selected tape be cleared so the operation can continue. All tapes used to perform the save operation should be initialized before the save command is issued.

DTACPR

Specifies whether data compression is performed.

***DEV:** If the tape device has the hardware compression feature installed, processing proceeds as if DTACPR(*YES) is specified. If the compression feature is not installed or if save data is written to a diskette or save file, processing proceeds as if DTACPR(*NO) is specified.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device data compression are performed if supported on the device.

***NO:** No data compression is performed.

***YES:** If the target device has the hardware compression feature, hardware compression is performed. If the feature is not present, software compression is performed.

COMPACT

Specifies whether device data compaction is performed.

***DEV:** Device data compaction is performed if the data is saved to tape and the target device supports the compaction feature.

Note: If *DEV is specified on both the DTACPR parameter and the COMPACT parameter, only device data compaction is performed if compaction is supported on the device. Otherwise, data compression is performed if supported on the device.

If *YES is specified on the DTACPR parameter and *DEV is specified on the COMPACT parameter, both device data compaction and device data compression are performed if supported on the device.

***NO:** Device data compaction is not performed.

Examples

Example 1: Tapes Cleared Automatically

```
SAVSYS  DEV(TAP01)  CLEAR(*ALL)
```

This command saves licensed internal code, system objects, all user profiles (including private authority for objects), and all line, controller, and device descriptions. They are saved on the tape put on the TAP01 tape drive. Each uncleared tape is automatically cleared when it is encountered, and the save operation continues without operator intervention.

Example 2: Operating Receives Message of Exceeded Storage Capacity

```
SAVSYS  DEV(TAP01)  VOL(ABCDE)
```

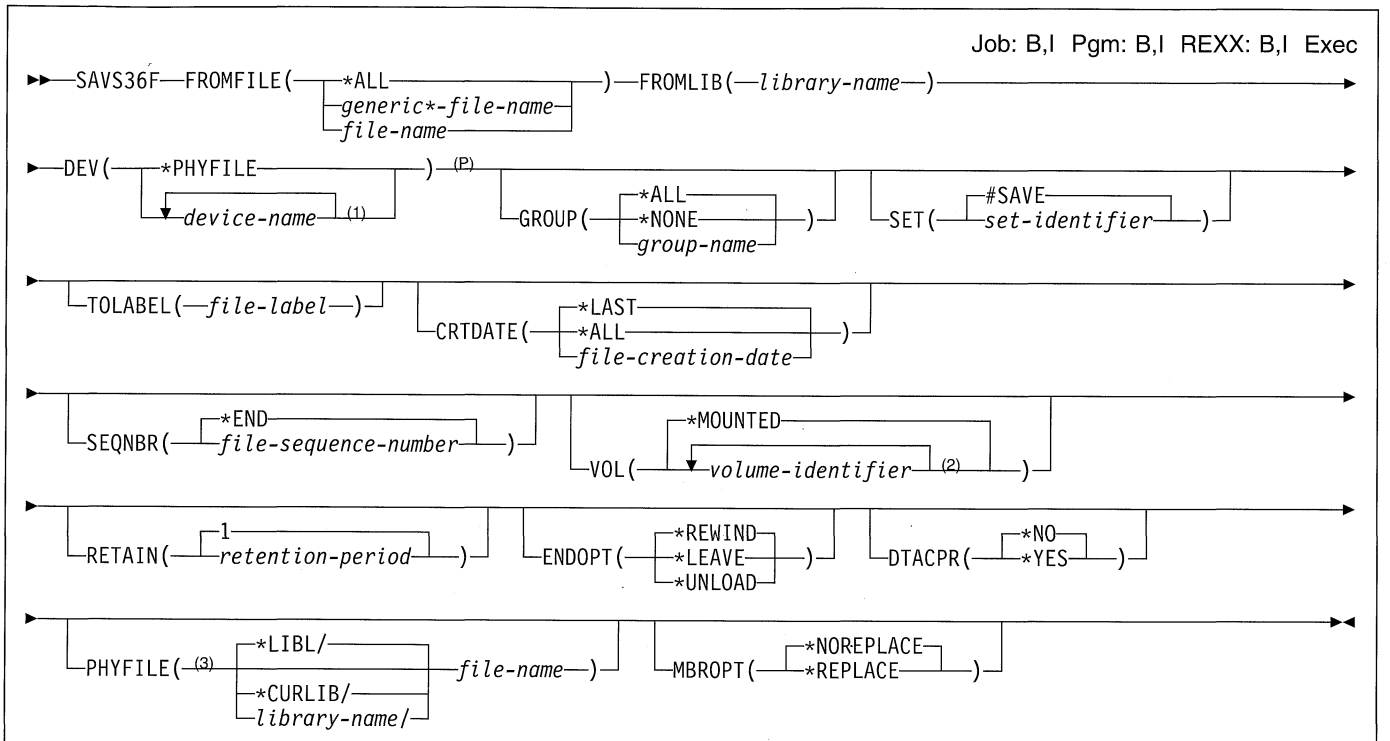
The system data is saved on the TAP01 tape drive, starting on the tape volume labeled ABCDE. If the save operation exceeds the storage capacity of one tape, a message requesting that another volume be put on the TAP01 tape drive is shown to the operator.

Example 3: Saving Data on Two Tape Drives in Alternating Order

```
SAVSYS  DEV(TAP01 TAP02)
```

The system data is saved on tape drives TAP01 and TAP02 in alternating order. If the save operation exceeds the storage capacity of two tapes, a message requesting that another volume be put on TAP01 is shown to the operator. The tapes are rewound at the completion of the save system operation.

SAVS36F (Save System/36 File) Command



Notes:

1 A maximum of 4 repetitions can be specified if the device named is a tape device. Repetitions are not valid for a diskette device.

P All parameters preceding this point can be specified in positional form.

2 A maximum of 50 repetitions

3 PHYFILE required if DEV(*PHYFILE) is specified.

Purpose

The Save System/36 File (SAVS36F) command saves the following:

- A copy of a single database physical or logical file to diskette, tape or another database physical file in the same format as if a System/36 had saved the file.
- A copy of multiple database physical or logical files to diskette or tape in the same format as if a System/36 had saved the files as a **Save All Set**. The following groups of files can be saved together with one operation:
  - All files in a library
  - All files in a specific **file group**
  - All files that are not part of a file group
  - All files that begin with a specified set of characters

A **Save All Set** is a group (set) of files that share the same group (set) name and that are saved (copied) to diskette or tape with one operation. The set of files can be restored (copied back from diskette or tape) with a single operation by referring to the set name (see the Restore System/36 Files (RSTS36F) command).

File groups are defined by file names that contain a period. The characters preceding the period identify the file group, and the characters following the period identify the file within the group. As with file names within the System/36 environment, the maximum number of characters is eight, including the period. Files with names that do not contain a period are not part of a file group. The following examples show the names of files within a file group.

```
PAYROL.A
PAYROL.B    Files in File Group PAYROL
PAYROL.C

A.ACCTS
A.INV
A.PROL      Files in File Group A
A.B.GO
A.B.INV

A.B.GO
A.B.INV     Files in File Group A.B
```

The saved files can be restored to the following systems:

- System/36 (RESTORE procedure or $COPY utility)
- AS/400 (Restore System/36 File (RSTS36F) command)

The SAVS36F command is intended for exchanging files with a System/36. For creating a backup version of a file, the AS/400 save commands (for example, Save Object (SAVOBJ) or Save Changed Object (SAVCHGOBJ)) should be used.

Restrictions:

1. The following authorities are required (normally only applies when running on a system using resource security):
   - *USE authority for this command.
   - *USE authority for the file or group of files specified in the FROMFILE parameter.
   - *USE authority for the library specified in the FROMLIB parameter.
   - *CHANGE authority to the file specified on the PHYFILE parameter if saving to an existing physical file.
   - *USE authority for the library specified in the PHYFILE parameter if saving to a physical file.
   - *CHANGE authority for the library specified in the PHYFILE parameter if saving to a physical file and the file does not exist.
   - *USE authority for the diskette device description object, *USE authority for device file QSYSDKT, in library QSYS if saving to diskette.
   - *USE authority for the tape device description object, *USE authority for device file QSYSTAP, in library QSYS if saving to tape.
   - *USE authority for the based-on physical file if saving a logical file.
2. All diskettes that are used for the save operation should be initialized using the INZDKT CL command or the equivalent System/36 environment function (INIT operator control language (OCL) procedure or $INIT SSP utility). For a two-sided diskette, use a sector size of 256 or 1024. For a one-sided diskette, use a sector size of 128 or 512. If tape is used, each tape volume used should have been initialized with standard labels using the INZTAP CL command or the equivalent System/36 environment function (TAPEINIT OCL procedure or $TINIT SSP utility). Use a density of 1600 bits per inch when initializing the tape.

   Note: If the tape or diskette has not been initialized as stated above, the System/36 will not be able to process the media.
3. Object-level and record-level functions, other than read operations, should not be attempted for a file being saved by SAVS36F. Concurrent activity against the file (for example, moving the file or adding or removing records) can cause:
   - For a save operation of a single file (FROMFILE(file-name)), the save operation will end with escape message CPF9826 because the file cannot be allocated.
   - For a save operation of multiple files (FROMFILE(*ALL or generic*-file-name)), the save function sends an inquiry message CPA2C6A because the file cannot be allocated. The message allows an ignore, retry and cancel response. The ignore response bypasses this file and attempts to save the next file selected.
4. When saving a single file to diskette, the diskette cannot already contain an active file with the same label and creation date as the new file to be created.
5. When saving multiple files to diskette, the diskette used for the save cannot contain any active files.
6. Not all physical and logical files can be saved with the SAVS36F command.
   - Only logical files created under the System/36 environment (for example, through the BLDINDEX OCL procedure) or through a DDM request from a System/36 system can be saved. These files are saved as System/36 alternative index files.
   - All physical files created under the System/36 environment (for example, through the BLDFILE OCL procedure) or through a DDM request from a System/36 system are saved using information stored within the AS/400 file description. These files are saved as System/36 sequential, direct, or indexed physical files.
   - Any physical files created by AS/400 commands or utilities can be saved as long as the record length is not greater than 4096. These files are saved as System/36 sequential files.
7. To generate a save format which can be processed by the System/36 RESTORE procedure, the following information is not saved:
   - If saving a logical file, only the description of the file is saved. The index (or access path) is not saved.
   - If saving an indexed (keyed) physical file, the data is saved but the index is not. The index will be rebuilt after the file is restored.
8. The following restrictions apply to naming standards:
   - When saving a single file, the specified name (FROMFILE parameter) must meet naming standards. If not, message CPF0001 is sent when the SAVS36F command is processed.
   - If a file name is found during a save operation of multiple files (FROMFILE(*ALL or generic*-file-name)) that does not meet the System/36 naming standards, diagnostic message CPF2C0E is sent and the file is not saved.
9. Multiple files (FROMFILE(*ALL) or FROMFILE(generic*-name)) cannot be saved to a physical file.

Required Parameters

FROMFILE

Specifies the name of one or more files to save. The files being saved must be in the library specified in the FROMLIB parameter.

***ALL:** All files in the specified library are saved. The GROUP parameter is used to further describe which files are saved.

generic-file-name:* Specify the generic name of the file. A generic name is a character string of one or more

characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be saved only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

file-name: Specify the name of a single file to save.

FROMLIB

Specifies which library contains the database files to be saved.

DEV

Specifies the names of the devices used for the save operation. Each device name other than *PHYFILE must already be known on the system by a device description.

***PHYFILE:** Specifies a database physical file is to receive the copied file. The qualified name of the physical files must be specified on the PHYFILE parameter.

*PHYFILE is not valid when saving multiple files by specifying *ALL or generic*-file-name on the FROMFILE parameter.

If the physical file does not exist, it is created as a non-keyed, program-described file with a record length of 256. If a file already exists by this name, it is used as long as it is a non-keyed physical file with a record length of 256.

The copied records are put in the first member of the physical file. If the file has no members, a member is created using the name syntax 'Myymmdd' and the system date (for example, if the SAVS36F was run on June 28, 1987, the member name would be M870628).

device-name: Specify the name of the diskette device or the names of tape devices that are used for the operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. When more than one tape volume is used, using more than one tape device permits one tape volume to be rewound or unloaded while another tape device processes the next tape volume. More information on device names is in the *APPC Programmer's Guide*.

Optional Parameters

GROUP

Specify which file groups are to be saved. This parameter is only valid if *ALL is specified for the FROMFILE parameter.

***ALL:** All files are saved.

***NONE:** Only files that do not belong to a file group are saved.

group-name: Specify the name of a file group to be saved. Do not specify either the period (.), which indicates a file name, or the file name itself. For example, to save files belonging to the file group that includes PAYROL.A, PAYROL.B, and PAYROL.C, enter PAYROL for this parameter. Files that do not belong to the specified file group are not saved. A group-name can be up to seven characters long. The first character in the name must begin with an alphabetic character (A through Z, #, @ or $). The remaining characters can be any combination of characters (numeric, alphabetic and special). Commas(,), apostrophes('), quotation marks("), question mark(?), asterisks(*), and blanks are not allowed.

SET

Specify the set identifier used to identify the entire set of files to be saved. This parameter is only valid if *ALL or a generic-name is specified on the FROMFILE parameter.

#SAVE: The files are saved as a Save All Set with a set identifier of #SAVE.

set-identifier: Specify the set identifier that is used to identify the entire set of file to be saved. Valid values range from 1 through 8 characters in length. The first character in the name must begin with an alphabetic character (A through Z, #, @ or $). The remaining characters can be any combination of characters (numeric, alphabetic and special). Commas(,), apostrophes('), quotation marks("), question marks(?), asterisks(*), and blanks are not allowed.

TOLABEL

Specifies the label given to the new tape or diskette file created by the save of a single file.

The TOLABEL name is also put into the System/36 file descriptor record which is stored at the beginning of the tape or diskette file. When the file is restored on a System/36, its name is the TOLABEL value (unless it is overridden by a //FILE OCL statement). Up to 8 characters can be used for the label.

If no value is specified, the value of the FROMFILE parameter is used as the diskette or tape label. If the FROMFILE parameter is an extended name, the characters between the quotation marks are used as the label. For example, specifying FROMFILE("MIKE&BOB") and no TOLABEL parameter would be the same as specifying TOLABEL('MIKE&BOB').

Note: This parameter cannot be specified if *ALL or generic*-name is specified on the FROMFILE parameter. The name of each file saved is used for the diskette or tape file label. If the name of the file is an extended name, the characters between the quotation marks are used as the label. For example, if the name of the file being saved is "A+B," the name of the diskette or tape file would be A+B.

CRTDATE

Specifies, for a date-differentiated file (maintained by the System/36 environment), which instance (member) of the file is saved. The default is to save the last instance (most recently created member) of the specified file. The date must be typed in the job date format (DATFMT). If separators, specified by the job date separator value (DATSEP), are used, the value must be enclosed in apostrophes. This parameter is valid only if a database physical file is being saved.

***LAST:** The most recently created member for the specified file is saved.

***ALL:** All members in the date-differentiated file are saved. If the file being saved is not date-differentiated, only the last member created in the file is saved. *ALL is valid only when saving multiple files and either *ALL or a generic file name is specified on the FROMFILE parameter.

file-creation-date: Specify the creation date of the date-differentiated file member to be saved. A file creation date can only be specified if a single file is to be saved and a file-name is specified on the FROMFILE parameter. (The date must be entered in the job date format (DATFMT); if separators, specified by the job date separator value (DATSEP), are used, the value must be enclosed in apostrophes.)

The date specified is converted to year/month/day format and the member of the file (specified by the FILE parameter) with the name 'Myymmdd' is saved. For example, to save the instance of file 'FRED' created on July 10, 1986, specify FROMFILE(FRED) CRTDATE('07/10/86'), which would save member M860710 in file FRED.

This example allows a migrated System/36 user to identify the desired file/member very much like a System/36. However, it does not lend itself to letting the user save off a single member from an AS/400 system multi-member file. For example, the user who has a file named FRED with members BOB and JOE could not pick which member should be saved (the user would have to copy the member to a new file and save the new file specifying CRTDATE(*LAST) or rename the member to follow the 'Myymmdd' name syntax).

SEQNBR

Specifies, only when tape is used, which sequence number is used for the save operation.

Note: If multiple files are to be saved, the next file is saved to a file after the first file saved, and so on.

***END:** The system saves the specified file after the last sequence number on the tape.

If this sequence number already exists on the tape volume, the tape label at that sequence number must match the TOLABEL parameter. The existing file at that sequence number is overwritten, and all subsequent files on the volume are not accessible after the save.

If a new tape file is added to the tape, the sequence number must be one higher than the sequence number of the last tape file on that volume. No gaps are allowed in the series of sequence numbers.

If multiple files are being saved, this sequence number is used for the first file. All remaining files are saved as if *END was specified on the parameter SEQNBR.

file-sequence-number: Specify the sequence number of the file that is used.

VOL

Specifies the volume identifiers of the volumes on which the data is saved. The volumes must be placed in the device in the order specified on this parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The file to be saved is copied on whatever volume is on the device. For diskettes, use whatever volume is loaded in the diskette drive. If the diskette loaded in the diskette drive is full of active files, the save ends.

volume-identifier: Specify the volume identifiers of the tapes or diskettes used for saving the file. For each tape or diskette volume name, up to six characters can be specified using any combination of letters and numbers. The user should ensure that the volume name meets the System/36 volume identifier requirements. Up to 50 volume identifiers can be specified.

RETAIN

Specifies the retention period for the newly created tape or diskette file. The file is protected and cannot be written over until the day after the retention period ends.

1: The default retention period is one day.

retention-period: Specify the number of days the tape or diskette file should be kept. Valid values range from 0 through 999. If a retention period of 999 is specified, the tape or diskette file becomes a permanent file.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

Note: This parameter is ignored if a diskette device is specified in the DEV parameter.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

PHYFILE

Specifies the qualified name of the file that receives the copied source file member data. This parameter is required if DEV(*PHYFILE) is specified.

If a file by this name does not exist, it is created as a non-keyed, program-described physical file with a record length of 256 bytes. If a file already exists by this name, it is used as long as it is a non-keyed physical file with a record length of 256 bytes.

The copied records are put in the first member of the physical file. If the file has no members, a member is created using the name syntax 'Myymmdd' and the system date. For example, if the SAVS36LIBM was run on June 28, 1987, the member name would be M870628.

The name of the physical file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the physical file.

DTACPR

Specifies, when a diskette is used, whether the data is compressed into System/36 compatible format before it is written to the diskette. If the save command is operating while other jobs on the system are active and data compression is used, the overall system performance may be affected. This parameter is not valid if *PHYFILE or a tape device is specified on the DEV parameter.

***NO:** The data is not compressed before being written to the diskette.

***YES:** The data is compressed before being written to the diskette.

MBROPT

Specifies whether the new records replace or are added to the existing records.

***NOREPLACE:** If a file already exists by the name specified on the PHYFILE parameter in the specified library, an error message is sent and the data in that member is not replaced.

***REPLACE:** The system clears the existing member and adds the new records.

Examples

Example 1: Saving a Single File

```
SAVS36F  FROMFILE(PETE)  FROMLIB(QS36F)  DEV(I1)
```

This command saves the file named PETE located in library QS36F. Assuming that I1 is the name of a diskette device description, the file is saved on the diskette placed in the diskette drive. The diskette file label is PETE (same as the FROMFILE name). If PETE is a date-differentiated physical file, the most recently created instance (member) of PETE is saved. The diskette file has a retention period of one day (the retention period ends at midnight of the following day).

Example 2: Saving a Single File

```
SAVS36F  FROMFILE(MSTRPAY)  FROMLIB(PAYLIB)  DEV(T1 T2)
  TOLABEL('PAY.MSTR')  RETAIN(999)
```

The file named MSTRPAY located in library PAYLIB is saved. Assuming that T1 and T2 are tape devices, the file is copied to the tapes on devices T1 and T2. The tape file label is PAY.MSTR and the tape file is a permanent file. The last tape used for the save is rewound at the end of the save operation.

Example 3: Saving Multiple Files

```
SAVS36F  FROMFILE(*ALL)  FROMLIB(QS36F)  DEV(T1 T2)
  GROUP(*ALL) SET(ALLFL) RETAIN(999)
```

All database physical and logical files in library QS36F (including all files that belong to a file group) are saved. If any of the files are date-differentiated files, only the last member created in each file is saved. Assuming that T1 and T2 are tape devices, the files are copied to the tape volumes that are placed in tape drives T1 and T2. The label of the tape files created are the same as the names of the files that are saved. The first tape file created is located after the last sequence number on the tape. The remaining files are located after that first file. The tape files created are permanent. The last tape used for the save is rewound at the end of the save operation. The set identifier associated with this save all set is ALLFL.

Example 4: Saving Multiple Files

```
SAVS36F  FROMFILE(*ALL)  FROMLIB(QS36F)  DEV(T1 T2)
  GROUP(*NONE) CRTDATE(*LAST) SET(NOGFL) RETAIN(999)
```

All database physical and logical files in library QS36F except those files that belong to a file group are saved. If any of the files are date-differentiated files, only the last member created in each file is saved. Assuming that T1 and T2 are tape devices, the files are copied to the tape volumes that are placed in tape drives T1 and T2. The label of the tape files created is the same as the names of the files that are saved. The first tape file created is located after the last sequence number on the tape. The remaining files are located after that first file. The tape files created are permanent. The last tape used for the save is rewound at the end of the save operation. The set identifier associated with this save all set is NOGFL.

Example 5: Saving Multiple Files

```
SAVS36F  FROMFILE(*ALL)  FROMLIB(GRPLIB)  DEV(I1)
  GROUP(GRP) CRTDATE(*ALL)
```

All database physical and logical files in library GRPLIB that belong to file group GRP (GRP.01, GRP.02, and so on) are

saved. If any of the files are date-differentiated files, all members in the files are saved. Assuming that I1 is a diskette drive, the files are copied to the diskette that is placed in the diskette drive. The label of the diskette files created is the same as the names of the files that are saved. The diskette files expire after one day. The set identifier associated with this save all set is #SAVE.

Example 6: Saving Multiple Files

```
SAVS36F  FROMFILE(PAY*)  FROMLIB(PAYROLL)  DEV(I1)
  SET(PAYSET) CRTDATE(*LAST) VOL(PAYDKT) RETAIN(10)
```

All database physical and logical files in library PAYROLL whose names begin with the characters PAY (PAY.01, PAYRATE, and so on) are saved. If any of the files are date-differentiated files, only the last member created is saved. Assuming that I1 is a diskette drive, the files are copied to a diskette with a volume identifier of PAYDKT. The label of the diskette files created is the same as the names of the files that are saved. The diskette files expire after ten days. The set identifier associated with this save all set is PAYSET.

Additional Considerations

A status message (CPI2C11) is sent when the processing begins for each database physical file that is saved. A completion message (CPC2C14 or CPC2C15) is sent after each file has been successfully saved.

Saving a Single File

A single file can be saved by specifying the name of the file on the FROMFILE parameter. If the file has multiple members (date-differentiated file), the user can:

- Specify which instance of the file (member) is to be saved by specifying the file's creation date on the CRTDATE parameter.
- Specify that the most recently created file (last member) is to be saved by specifying *LAST on the CRTDATE parameter.

Note: *ALL cannot be specified on the CRTDATE parameter to save all instances of the file (all members) when saving a single file.

If the file is being saved to diskette or tape, the TOLABEL parameter can be used to specify the label of the new diskette or tape file created during the save operation. The label is also used when the file is restored.

If a single file is to be saved and a problem occurs that will not allow the operation to continue, an escape message is sent to the program message queue previous to the command processing program (CPP). This escape message informs the user of the cause and recovery of the problem. The message ends the operation.

Saving Multiple Files

Multiple files can be saved by specifying one of the following on the FROMFILE parameter:

- *ALL specifies the following:
  - If the value on the GROUP parameter is *ALL, all physical and logical files in the specified library (FROMLIB parameter) are saved, including all files that are part of file groups.
  - If the value on the GROUP parameter is *NONE, all physical and logical files in the specified library (FROMLIB parameter) are saved, except those files that are part of file groups.
  - If the value on the GROUP parameter is a file group name, all physical and logical files in the specified library (FROMLIB parameter) that are part of the specified file group are saved.
- generic*-file-name specifies that all physical and logical files in the specified library (FROMLIB parameter) with the same prefix as the generic name are saved.

When saving multiple files, a set identifier (SET parameter) is required. If a value is not specified, **#SAVE** is used.

The TOLABEL parameter cannot be specified. The label of the diskette or tape file created will be the same as the database file that is copied into it.

A specific file creation date cannot be specified on the CRTDATE parameter. If the file has multiple members (date-differentiated file), the most recently created member (CRTDATE(*LAST)) or all members (CRTDATE(*ALL)) can be saved.

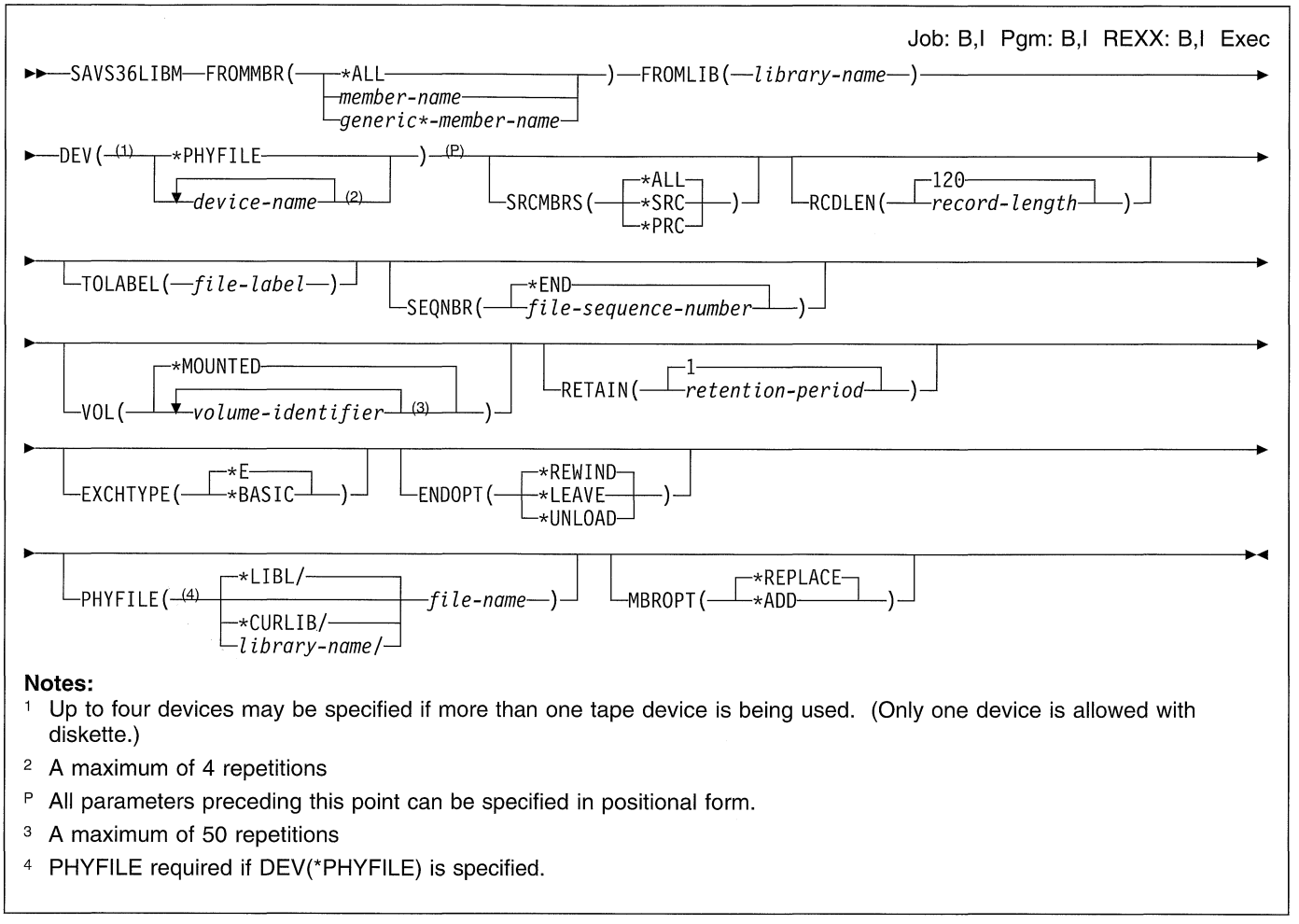
If multiple files are to be saved and a problem is found that will not allow the operation to continue for a particular file, an inquiry message will be sent. If the SAVS36F job is interactive, the message is sent to the work station where the job was submitted. If the job is batch, the message is sent to the QSYSOPR message queue. The inquiry message informs the user of the cause and recovery of the problem. The messages have the following response options:

- C–Cancel the save operation.
- R–Retry saving the file after the problem has been corrected.
- I–Ignore saving this file and continue with the save of the next file.

The default response is I.

If multiple files are to be saved, and the operation was successful for all selected files, a completion message (CPC2C19) is sent after all files have been saved. This completion message informs the user the number of files that were saved. If any of the selected files cannot be saved, an escape message (CPF2C5B) is sent at the end of the save operation. The message informs the user of the number of files that were saved and the number that were not saved.

SAVS36LIBM (Save System/36 Library Members) Command



Notes:

1 Up to four devices may be specified if more than one tape device is being used. (Only one device is allowed with diskette.)

2 A maximum of 4 repetitions

| P All parameters preceding this point can be specified in positional form.

3 A maximum of 50 repetitions

4 PHYFILE required if DEV(*PHYFILE) is specified.

Purpose

The Save System/36 Library Members (SAVS36LIBM) command creates a copy of source file members in a file on diskette or tape that can be restored on a System/36, or into a database physical file on the AS/400 system that can be sent to a System/36.

The saved member file is formatted like a record-mode LIBRFILE created on a System/36 using the System/36 FROMLIBR system OCL procedure (or the equivalent OCL call of the $MAINT SSP utility). On a System/36, the diskette, tape, or database physical file can be restored using the TOLIBR system OCL procedure (or the equivalent OCL call of the $MAINT SSP utility).

If saving to a database physical file but the specified file (PHYFILE parameter) does not exist, it is created.

This command is intended only for exchanging source and procedure data with a System/36. It provides a simplified command interface for an AS/400 system customer who migrated from a System/36, but is not well-suited for backing up of AS/400 system source files. Use the AS/400 system CL commands (SAVOBJ or SAVCHGOBJ) for creating a backup copy of an AS/400 system source file or individual source file members.

Restrictions:

1. The following authorities are required when running on a system using resource security:
   - *USE authority for this command
   - *USE authority for the library specified in the FROMLIB parameter
   - *USE authority for file QS36SRC in the specified library if saving source library members
   - *USE authority for file QS36PRC in the specified library if saving procedure library members
   - *USE authority for the library specified in the PHYFILE parameter if saving to a physical file
   - *CHANGE authority for the library specified in the PHYFILE parameter if saving to a physical file and the file does not exist
   - *CHANGE and *OBJMGMT authority for that file if saving to a physical file with MBROPT(*ADD)
   - *ALL authority for the file if saving to a physical file with MBROPT(*REPLACE)

- *USE authority for the diskette device description object, *USE authority for device file QSYSDKT in library QSYS if saving to diskette
- *USE authority for the tape device description object and *USE authority for device file QSYSTAP in library QSYS if saving to tape

2. All diskettes that are used to save the members should be initialized using the INZDKT CL command or the equivalent System/36 environment function (INIT OCL procedure or $INIT SSP utility). For a two-sided diskette, use a sector size of 256 or 1024. For a one-sided diskette, use a sector size of 128 or 512. If tape is used, each tape volume used should first be initialized with standard labels using the INZTAP CL command or the equivalent System/36 environment function (TAPEINIT OCL procedure or $TINIT SSP utility). Use a density of 1600 bits per inch when initializing the tape.

   Note: If the tape or diskette has not been initialized as stated above, the System/36 will not be able to process the media.

   If a tape or diskette is used that has not been properly initialized, a message is sent to the system operator that allows the operator to cancel the save or initialize the volume and continue.

3. Object-level functions, other than read operations, should not be used for files QS36SRC and QS36PRC while members are being saved by SAVS36LIBM. If the necessary files cannot be allocated, no members are saved.

   Record-level functions, other than read operations, should not be used for members being saved. Concurrent activity against a member (for example, adding or removing records) can cause the member to be omitted from the save operation.

4. If saving a file to diskette, the diskettes used cannot contain an active file with the same name as the TOLABEL parameter value, because the AS/400 system does not allow duplicate diskette file labels.

5. Only members from source files QS36SRC (for *SRC members) and QS36PRC (for *PRC members) in the specified library can be saved using the SAVS36LIBM command. Only the member data is saved from the source file member (that is, the source sequence number and change date fields are not saved).

6. The specified member name or generic member name (FROMMBR parameter) must meet AS/400 system naming standards. When saving a member that has an extended name, the quotation mark characters are not stored as part of the member name in the output file. For example, member "A+B" would be saved as A+B.

Required Parameters

FROMMBR

Specifies the names of the members being saved. Specify a single member name or a single generic member name or *ALL for the members to be saved.

***ALL:** All members are saved of the member type specified on the SRCMBRS parameter.

member-name: Specify the names of the members being saved.

generic-member-name:* Specify the generic name of the member. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be saved only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

FROMLIB

Specifies which library contains the members being saved.

DEV

Specifies the names of the devices used for the save operation. Each device name must already be known on the system by a device description.

***PHYFILE:** The output file is the database physical file specified by the PHYFILE parameter.

device-name: Specify the name of the diskette device or the names of tape devices that are used for the operation. If more than one tape device is used, specify the names of the devices in the order in which they are used. When more than one tape volume is used, using more than one tape device permits one tape volume to be rewound or unloaded while another tape device processes the next tape volume. More information on device names is in the *APPC Programmer's Guide*.

Optional Parameters

SRCMBRS

Specifies which member types (source and procedure members on System/36) are saved.

***ALL:** Save System/36 source and procedure members (from QS36SRC and QS36PRC) which match the member name specified (FROMMBR parameter). For example, if FROMMBR(PAY*) and SRCMBRS(*ALL) is specified, all members from both source files that begin with the letters 'PAY' are saved.

***SRC:** Only System/36 source members (from file QS36SRC), which match the member name specified (FROMMBR parameter), are saved.

***PRC:** Only System/36 OCL procedure members (from file QS36PRC), which match the member name specified (FROMMBR parameter), are saved.

RCDLEN

Specifies the file record length (in bytes) used when copying the members. This parameter is ignored when saving members to an existing database physical file.

120: This is the maximum record length allowed for System/36 source and procedure library members. Using this length ensures that no data is truncated when copying the records to the output file.

record-length: Specify the logical record length of the statements in the member. Valid values range from 40 through 120 and must be less than or equal to the record length of the file.

TOLABEL

Specifies the label of the diskette or tape file that receives the copied source file member data. Up to 8 characters can be used for the label value for the diskette or tape file. If the DEV parameter is not *PHYFILE, a TOLABEL value must be specified.

SEQNBR

Specifies, only when tape is used, which sequence number is used for the save operation.

***END:** The system saves the specified members after the last sequence number on the tape. If this sequence number already exists on the tape volume, the tape label at that sequence number must match the TOLABEL parameter value. The existing data file at that sequence number is overwritten, and all subsequent files on the volume are not accessible after the save operation.

If a new tape file is added to the tape, the sequence number must be one greater than the sequence number of the last tape file on that volume. No gaps are allowed in the series of sequence numbers.

file-sequence-number: Specify the sequence number of the file that is used.

VOL

Specifies the volume identifiers of the volumes on which the data is saved. The volumes must be placed in the device in the order specified on this parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***MOUNTED:** The members saved are copied on the volume that is in the device. For diskettes, use the volume that is in the diskette drive. If the diskette loaded in the diskette drive is full of active files, the save operation ends.

volume-identifier: Specify the volume identifiers of the tapes or diskettes used for saving the members. For each tape or diskette volume name, up to 6 characters can be specified using any combination of letters and numbers. The user should ensure that the volume name meets the System/36 volume identifier requirements. Up to 50 volume identifiers can be specified.

RETAIN

Specifies the retention period for the newly created tape or diskette file. The file is protected and cannot be written over until the day after the retention period ends.

1: A retention period of one day is used.

retention-period: Specify the number of days the tape or diskette file should be kept. Valid values range from 0 through 999. If a retention period of 999 is specified, the tape or diskette file becomes a permanent file.

EXCHTYPE

Specifies the exchange type for the newly created diskette file.

***E:** The default is to create an E-exchange diskette file. An E-exchange file is a system save file that can be restored either on a System/36 (using the TOLIBR procedure) or on an AS/400 system (using the RSTS36LIBM command).

***BASIC:** The output diskette file is to be in basic exchange format. A basic exchange format file can be restored or copied to a System/34 or System/32 or any other system that supports the basic exchange diskette format.

ENDOPT

Specifies the operation that is automatically performed on the tape volume after the operation ends. If more than one volume is included, this parameter applies only to the last tape volume used; all other tape volumes are rewound and unloaded when the end of the tape is reached.

Note: This parameter is ignored if a diskette device is specified in the DEV parameter.

***REWIND:** The tape is automatically rewound, but not unloaded, after the operation has ended.

***LEAVE:** The tape does not rewind or unload after the operation ends. It remains at the current position on the tape drive.

***UNLOAD:** The tape is automatically rewound and unloaded after the operation ends.

PHYFILE

Specifies the qualified name of the file that receives the copied source file member data. This parameter is required if DEV(*PHYFILE) is specified.

If a file by this name does not exist, it is created in the current library if a library name was not specified, as a non-keyed, program-described physical file with the specified record length (RCDLEN parameter). If a file already exists by this name, it is used as long as it is a non-keyed physical file with a record length from 40 to 120.

The copied records are put in the first member of the physical file. If the file has no members, a member is created using the name syntax 'Myymmdd' and the system date (for example, if the SAVS36LIBM was run on June 28, 1987, the member name would be M870628).

The name of the physical file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the file that receives the copied source file member data.

MBROPT

Specifies whether the new records replace or are added to the existing records.

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

Examples

Example 1: Saving Single Procedure Member

```
SAVS36LIBM  FROMMBR(XYZ1)  FROMLIB(JOHNSON)  DEV(I1)
  SRCMBRS(*PRC)  TOLABEL(XYZ1)
```

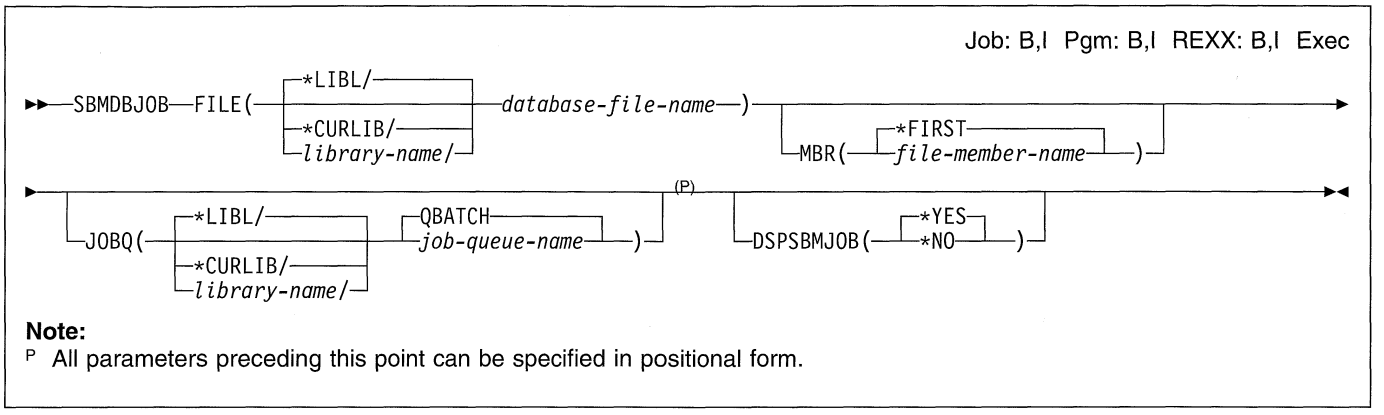
The single OCL procedure member XYZ1 (in source file QS36PRC) in library JOHNSON is saved. Assuming I1 is a diskette device, the member is saved to diskette file XYZ1. The file length is 120 and the retention period is one day.

Example 2: Saving All Source and Procedure Members

```
SAVS36LIBM  FROMMBR(X*)  FROMLIB(ORDER)  DEV(*PHYFILE)
  PHYFILE(NETLIB/S36SRC) MBROPT(*ADD)
```

All source and procedure members (members in QS36SRC and QS36PRC) with names starting with the character 'X' in library ORDER are saved into database physical file S36SRC in library NETLIB. The copied records are added after any records already in the file.

SBMDBJOB (Submit Database Jobs) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Submit Database Jobs (SBMDBJOB) command allows a job that is running to submit other jobs to job queues to be run as batch jobs. The input stream is read either from a physical database file or from a logical database file in single-record format. This command specifies the name of the database file and member from which the jobs are read, the name of the job queue to be used if the Batch Job (BCHJOB) command has JOBQ(*RDR) specified, and whether jobs being submitted can be displayed by the Work with Submitted Jobs (WRKSBMJOB) command.

A Submit Database Jobs operation reads the file once and ends when the end-of-file is read or when an End Input (ENDINP) command (a delimiter) is encountered. The ENDINP command (a delimiter) is not recognized if it is within an inline file that ends with characters that are not default ending characters (as specified in the ENDCHAR parameter of the Data (DATA) command). The SBMDBJOB operation can be canceled either by canceling the request from the system request menu or by canceling the job in which the process is running.

In contrast to a spool reader started with the Start Database Reader (STRDBRDR) command, the SBMDBJOB command operates in the same process as the requesting function and does not do syntax checking on the input stream.

Restriction: The specified database file either must consist of single-field records and must have an arrival sequence access path, or it must be a standard database source file.

Required Parameters

FILE

Specifies the name of the database file from which the input stream is read.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file.

Optional Parameters

MBR

Specifies the name of the member in the specified file that contains the input stream being read.

***FIRST:** The first member in the database file is used.

file-member-name: Specify the name of the member that contains the input stream being read.

JOBQ

Specifies the job queue on which the job entries are placed. A job entry is placed on this queue for each job in the input stream that has JOBQ(*RDR) specified on the Batch Job (BCHJOB) command. If *RDR is not specified on the BCHJOB command, the job queue specified on the BCHJOB command or in the job description is used. (The job queue for each job in the input stream can be different.) This parameter is valid only if ACTION(*SUBMIT) is specified on this command, in the existing network job entry, or in a subsequent Change Network Job Entry (CHGNETJOBE) command.

Note: If both the user identified in the job description of the job being read and the user processing the Submit Database Job (SBMDBJOB) command are not authorized to the job queue on which the job should be placed, the job ends and a diagnostic message is placed in the job log. The input stream, continues to be processed, starting with the next job. If either user is authorized to the job queue, the job runs without error.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QBATCH: The job entry is placed on the QBATCH job queue, which is the default job queue, if JOBQ(*RDR) is specified on the BCHJOB command.

job-queue-name: Specify the name of the job queue to which each job in the job stream is sent if JOBQ(*RDR) is specified on the BCHJOB command.

DSPSBMJOB
Specifies whether the jobs being submitted can be displayed on the submitted jobs display. Any submitted job of the type specified by the SBMFROM parameter of the WRKSBMJOB command can be displayed if *YES is specified on this parameter.

***YES:** This job can be displayed by the WRKSBMJOB command.

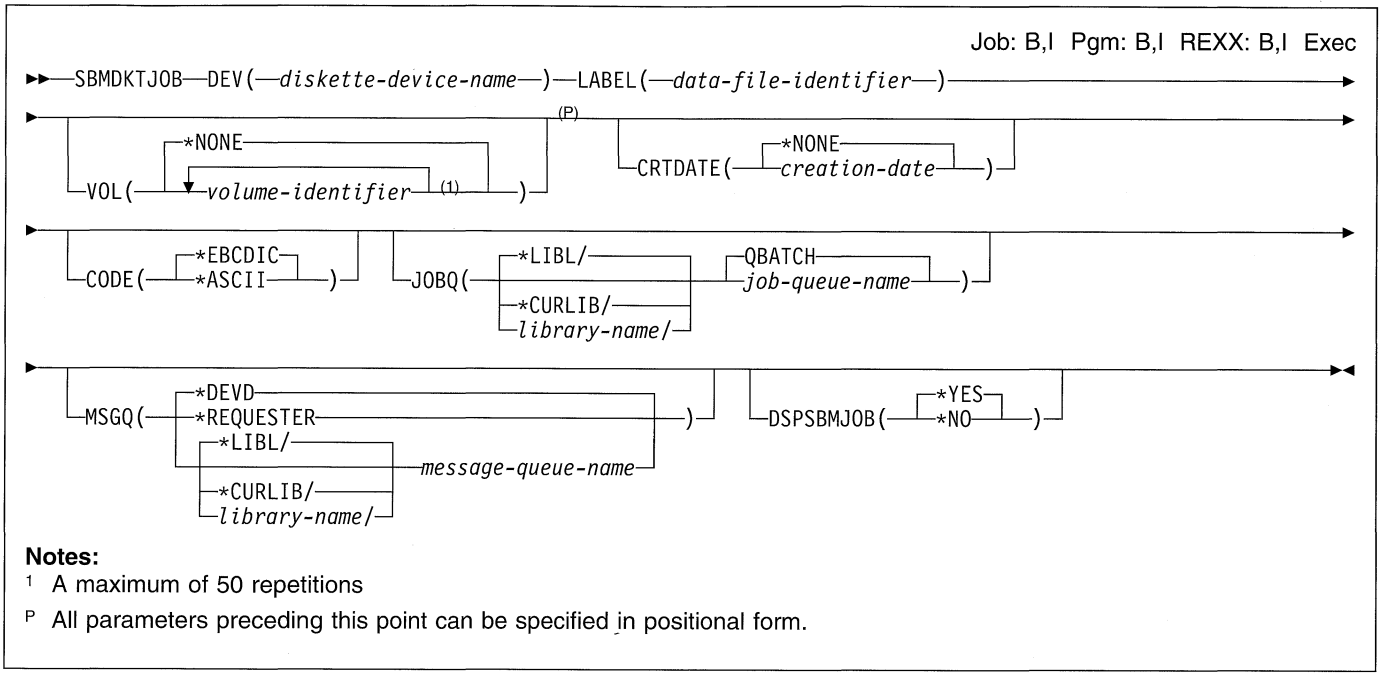
***NO:** This job is not displayed on any display produced by the WRKSBMJOB command.

Example

```
SBMDBJOB  FILE(QGPL/BILLING)
```

This command submits jobs using input from the database file named BILLING, which is in the QGPL library. The first member in the BILLING file contains the input stream to be processed. The default system-supplied job queue QBATCH is used.

SBMDKTJOB (Submit Diskette Jobs) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Submit Diskette Jobs (SBMDKTJOB) command allows jobs that are running to submit other jobs to job queues to be run as batch jobs. The input stream is read from a diskette. This command specifies:

- The name of the diskette device
- The label identifier
- The volume, creation date, and interchange code type of the file containing the input stream
- The names of the job queue and message queue used
- Whether jobs being submitted can be displayed by the Work with Submitted Jobs (WRKSBMJOB) command

A Submit Diskette Jobs operation reads the file once and ends when the end-of-file is read or when an ENDINP (End Input) command (delimiter) is encountered. The ENDINP command (delimiter) is not recognized if it is within an inline file that ends with characters that are not default ending characters (as specified in the ENDCHAR parameter of the DATA command). The SBMDKTJOB operation can be canceled either by canceling the request from the system request menu or by canceling the job in which the operation is running.

In contrast to a spool reader started with the Start Diskette Reader (STRDKTRDR) command, the SBMDKTJOB command operates in the same process as the requesting function and does not do syntax checking on the input stream.

Note: This command cannot be used to read the data files of diskettes that are in the save/restore format.

Required Parameters

DEV
Specifies the name of the diskette device used to read the input stream. Specify the name of the diskette device.

LABEL
Specifies the data file label (identifier) of the diskette data file on the diskette that contains the input stream. Specify the data file identifier. The data file identifier can be no longer than eight characters.

Optional Parameters

VOL
Specifies one or more volume identifiers used by the file. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***NONE:** No volume identifiers are specified. The current volume is used.

volume-identifier: Specify the identifiers of one or more volumes in the order that they are put on the diskette device and used. Each volume identifier contains up to six characters. A blank is used as the separator character when listing multiple identifiers.

CRTDATE
Specifies the creation date of the object.

Note: The creation date should not be specified if the date is not checked. If the date written on the diskette containing the data file does not match the date specified here, an error message is sent

to the message queue named in the MSGQ parameter.

***NONE:** The creation date is not specified, and no check is made.

creation-date: Specify the creation date of the data file to be read. The date should be in the format specified by the QDATFMT system value.

CODE

Specifies the character code used. The code can be either extended binary-coded decimal interchange code (*EBCDIC) or the American National Standard Code for Information Interchange (*ASCII).

***EBCDIC:** The extended binary-coded decimal interchange code (EBCDIC) character set code is used.

***ASCII:** The ASCII character code is used.

JOBQ

Specifies the job queue on which the job entries are placed. A job entry is placed on this queue for each job in the input stream that has JOBQ(*RDR) specified on the Batch Job (BCHJOB) command. If *RDR is not specified on the BCHJOB command, the job queue specified on the BCHJOB command or in the job description is used. (The job queue for each job in the input stream can be different.) This parameter is valid only if ACTION(*SUBMIT) is specified on this command, in the existing network job entry, or in a subsequent Change Network Job Entry (CHGNETJOBE) command.

Note: If both the user identified in the job description of the job being read and the user processing the Submit Diskette Job (SBMDKTJOB) command are not authorized to the job queue on which the job should be placed, the job ends and a diagnostic message is placed in the job log. The input stream, continues to be processed, starting with the next job. If either user is authorized to the job queue, the job runs without error.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QBATCH: The job entry is placed on the QBATCH job queue, which is the default job queue if JOBQ(*RDR) is specified on the BCHJOB command.

job-queue-name: Specify the name of the job queue to which each job in the job stream is sent if JOBQ(*RDR) is specified on its BCHJOB command.

MSGQ

Specifies the qualified name of the message queue to which messages are sent.

***DEVD:** The messages are sent to the message queue indicated by the device description of the device being used.

***REQUESTER:** The messages are sent to the message queue of the user who started the process. This value is not valid for batch jobs.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue to which operational messages are being sent.

DSPSBMJOB

Specifies whether the jobs being submitted can be displayed on the submitted jobs display. Any submitted job of the type specified by the SBMFROM parameter of the WRKSBMJOB command can be displayed if *YES is specified.

***YES:** This job can be displayed by the WRKSBMJOB command.

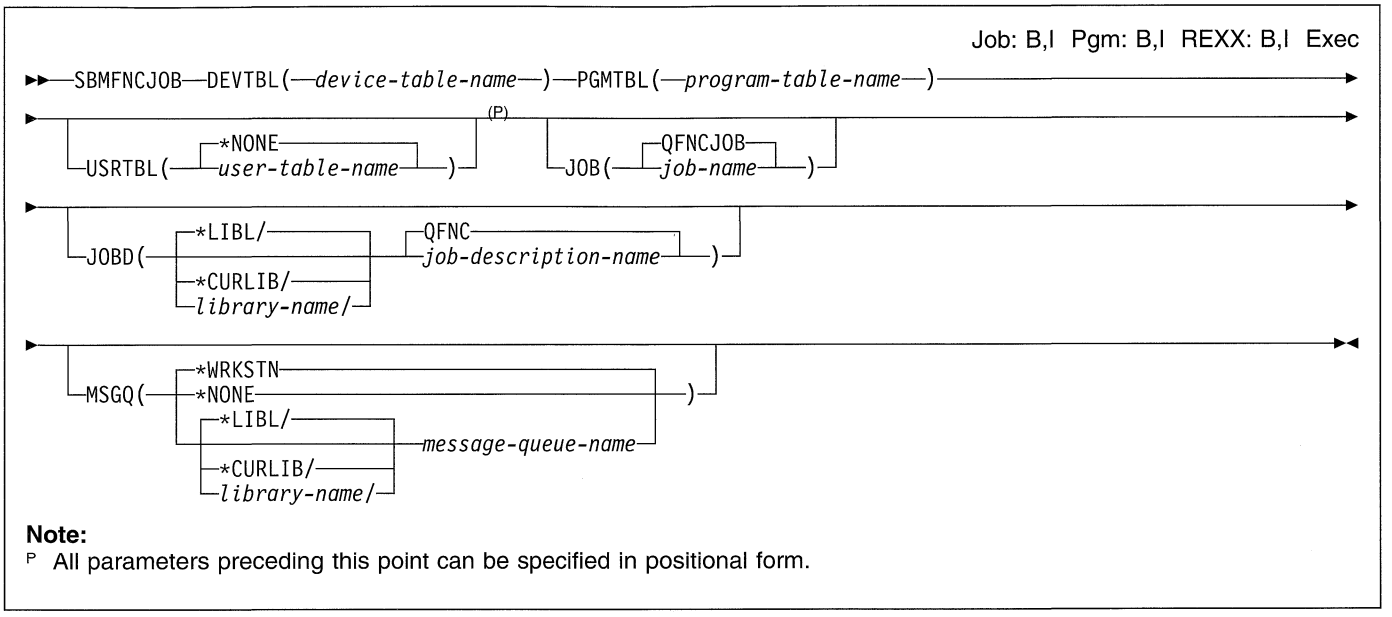
***NO:** This job is not displayed on any display produced by the WRKSBMJOB command.

Example

```
SBMDKTJOB  DEV(QDKT)  LABEL(OCT24)  VOL(SALES)
```

This command submits diskette jobs using diskette input from the device QDKT. The submit diskette jobs function gets its input from the data file named OCT24 with the volume identifiers SALES. The default job queue QBATCH is used as the receiving job queue when JOBQ(*RDR) is found in the job description. Operational messages are sent to the message queue defined by the device.

SBMFNCJOB (Submit Finance Job) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Submit Finance Job (SBMFNCJOB) command submits a batch job so the user's finance application programs communicates with the 4701 or 4702 control unit application programs.

Use the SBMFNCJOB command only if:

- Communicating with a 4701 or 4702 control unit
- A device table and a program table have been defined using the Work with Device Table (WRKDEVTBL) and Work with Program Table (WRKPGMTBL) commands; defining a user table using the Work with User Table (WRKUSRTBL) command is optional
- The user's 4701 or 4702 control unit application program sends data (transactions) first and expects a response
- The user's 4701 or 4702 control unit application program passes data in the proper format

Restriction: This command is shipped with public *EXCLUDE authority.

Required Parameters

DEVTBL

Specifies the name of the device table that the finance job uses to determine which 4704 or 3624 devices it controls.

PGMTBL

Specifies the name of the program table that the finance job uses to determine, from the program ID (sent in the data stream with a finance transaction), which system user program names will process the finance transaction.

Optional Parameters

USRTBL

Specifies the name of the user table that the finance job uses to verify a valid finance user when a finance sign-on is received.

***NONE:** No user IDs are verified.

user-table-name: Specify the name of a user table that defines user IDs for the 4700 device.

JOB

Specifies the job name that is associated with the submitted finance job. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

QFNCJOB: The job name is submitted as QFNCJOB.

job-name: Specify the job name that is associated with the submitted finance job.

JOBD

Specifies the qualified name of the job description that is used by the finance job.

The name of the finance job can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QFNC: The submitted finance job uses the job description QFNC.

job-description-name: Specify the name of a job description that is used by the finance job. (If no library name is given, the job description is found through the library list used by the job in which the SBMFNCJOB command is entered.)

MSGQ

Specifies the qualified name of the message queue to which messages are sent.

***WRKSTN:** The finance messages are sent to the message queue of the work station from which the finance job was submitted.

***NONE:** No finance messages are sent to a message queue.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue to which messages are sent. (If no library name is given, the library list of the job issuing the SBMFNCJOB command is used to find the queue.)

Examples

Example 1: Submitting a Batch Job that Communicates with Devices it Acquires

```
SBMFNCJOB  DEVTBL(DEVTBL1)  PGMTBL(PGMTBL1)
  USRTBL(USRTBL1)
```

This command submits batch job QFNCJOB. The job communicates with all devices it acquires from device table DEVTBL1, allowing users whose user IDs are found in USRTBL1 to sign on the devices. Each transaction sent by the finance devices is processed by determining, in PGMTBL1, which application program must be called, then it calls that program.

The job description used by the finance job in this example is QFNC.*LIBL. Messages sent as a result of the finance job are sent to the message queue of the work station from which the job was submitted.

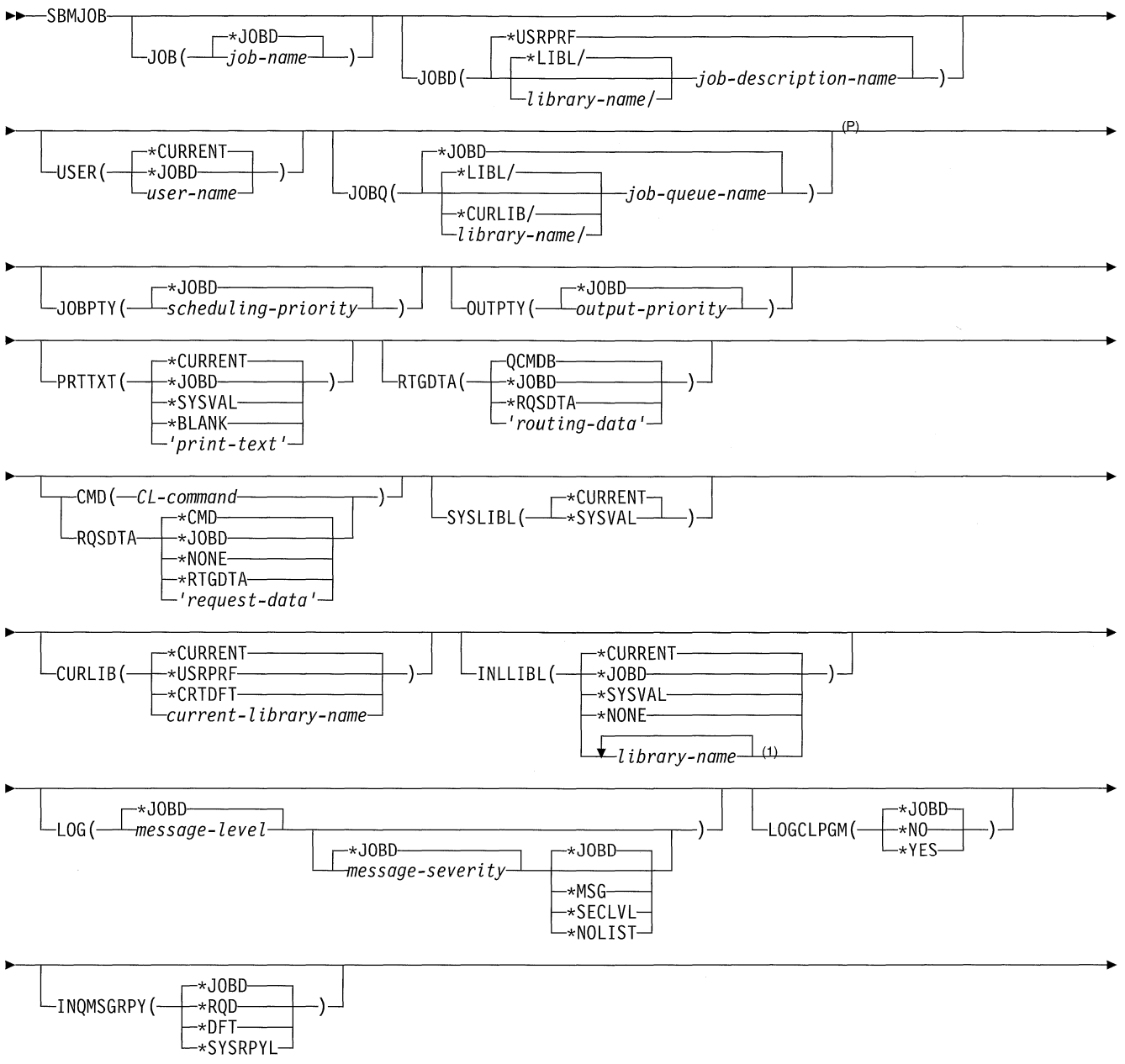
Example 2: User IDs Not Verified

```
SBMFNCJOB  DEVTBL(DEVTBL2)  PGMTBL(PGMTBL2)
  JOB(CTFJOB)  JOBD(CTFJOBD) MSGQ(*NONE)
```

This command submits batch job CTFJOB. CTFJOB runs under job description CTFJOBD and does not send messages to any work station message queue while running. No verification of user IDs is performed by the finance job.

SBMJOB (Submit Job) Command

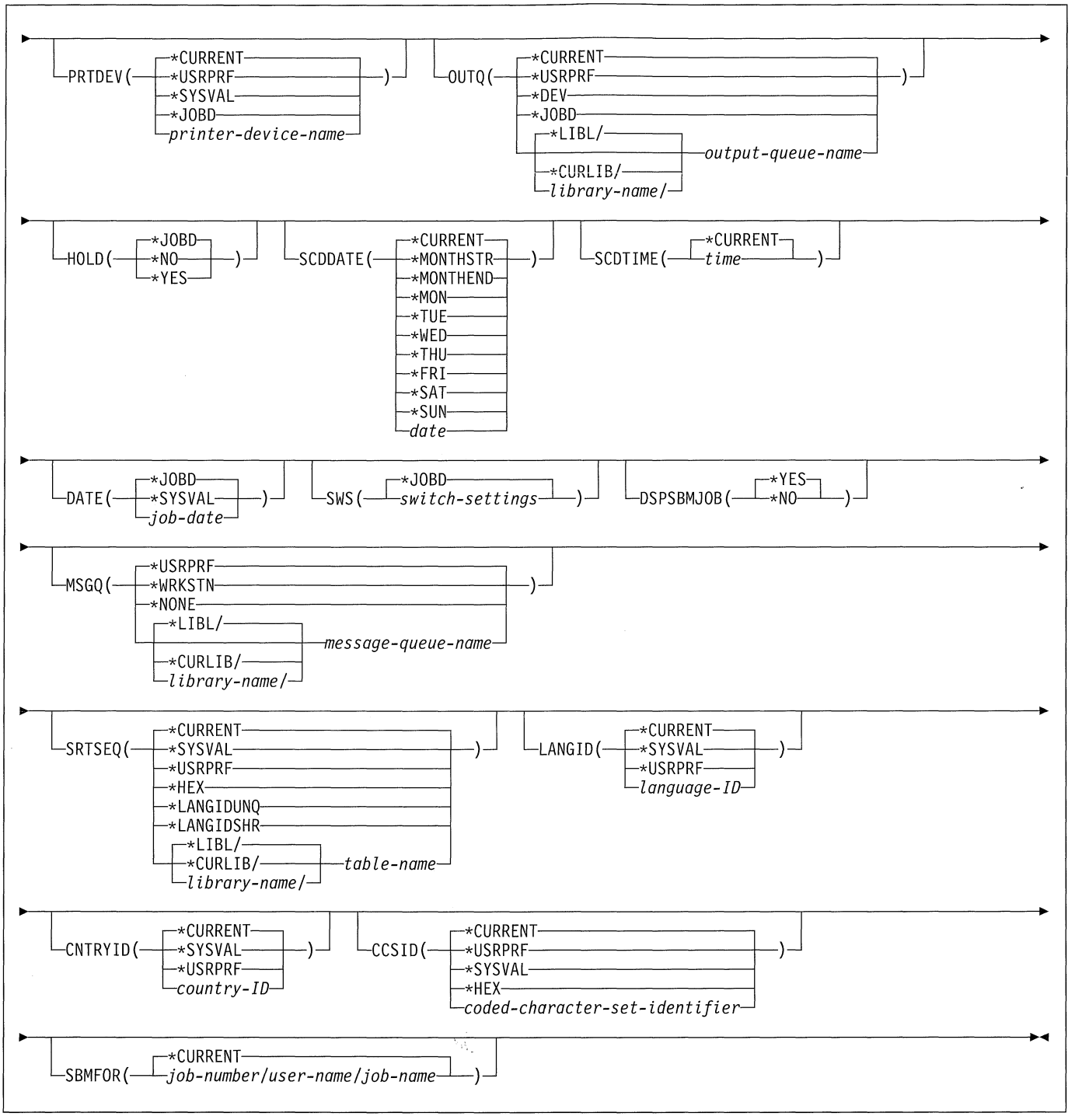
Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

P All parameters preceding this point can be specified in positional form.

1 A maximum of 25 repetitions



Purpose

The Submit Job (SBMJOB) command allows a job that is being run to submit another job to a job queue to be run later as a batch job. Only one element of request data can be placed in the new job's message queue. The request data can be a CL command if the routing entry used for the job specifies a CL command processing program.

Note: A job started by the SBMJOB command uses the accounting code of the job that submits the job. The accounting code specifications on the submitted jobs' JOBD and USRPRF parameters are ignored.

Optional Parameters

JOB

Specifies the job name that is associated with the submitted job while it is being processed by the system.

***JOBD:** The simple name of the job description used with this job is the name of the job itself.

job-name: Specify the simple name of the job that is used while it is being processed by the system.

JOBD

Specifies the job description used to submit jobs for batch processing. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***USRPRF:** The job description specified in the user profile under which the submitted job runs is used. The user profile is specified on the USER parameter.

The name of the job description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

library-name: Specify the name of the library to be searched.

job-description-name: Specify the name of the job description.

USER

Specifies the name of the user profile under which the job is submitted. USER(*JOBD) is not valid when USER(*RQD) is specified on the Create Job Description (CRTJOBD) command.

***CURRENT:** The user profile under which the current job is running is used.

***JOBD:** The user profile name in the specified job description is used for the job being submitted.

When the system is running under security level 40, the user of this command must be authorized to the user specified in the job description.

user-name: Specify the user profile name that is used for the submitted job. The user must be authorized to the user name; the user-name must be authorized to the job description (JOBD).

JOBQ

Specifies the qualified name of the job queue on which this job is placed.

***JOBD:** The submitted job is placed on the job queue named in the specified job description.

The name of the job queue name can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-queue-name: Specify the qualified name of the job queue on which the submitted job is placed.

JOBPTY

Specifies the scheduling priority for the submitted job. Valid values range from 1 through 9, where 1 is the highest priority and 9 is the lowest priority. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***JOBD:** The scheduling priority specified in the job description is used.

scheduling-priority: Specify a value, ranging from 1 through 9, for the scheduling priority for the job.

OUTPTY

Specifies the output priority for spooled files that are produced by the submitted job. The highest priority is 1 and the lowest priority is 9. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***JOBD:** The output priority specified in the job description is used for the job.

output-priority: Specify a value, ranging from 1 through 9, for the priority of the output files of the submitted job.

PRTTXT

Specifies up to 30 characters of text to be printed at the bottom of each page of output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***CURRENT:** The same print text of the submitting job is used.

***JOBD:** The value in the job description is used.

***SYSVAL:** The system value (QPRTTXT) is used.

***BLANK:** Text is not specified.

'print-text': Specify the character string that is printed at the bottom of each page. Up to 30 characters can be entered, enclosed in apostrophes.

RTGDTA

Specifies the routing data that is used with this job description to start jobs. The routing data is used to determine the routing entry (in the subsystem description) that identifies the program in which the job runs.

QCMDB: The routing data used by the IBM-supplied batch subsystem, QBATCH, to route batch jobs to the IBM-supplied control language processor QCMD is used.

***JOBD:** The routing data specified in the job description is used to start the routing step.

***RQSDTA:** Up to 80 characters of the request data specified in the RQSDTA parameter of this command is used as the routing data for the job.

'routing-data': Specify the character string that is used as the routing data for starting the job. Up to 80 characters can be entered enclosed in apostrophes, if necessary.

CMD

Specifies the command that runs in the submitted job. The IBM-supplied default routing program QCMD must be used when the job is started or the job will not run. Because the command you specify is used for the request data, the value specified on the RQSDTA parameter in the job description is ignored. The command can be a maximum of 3000 characters in length.

RQSDTA

Specifies the request data that is placed in the submitted job's message queue. For example, if RTGDTA (QCMDB) is specified, the IBM-supplied batch subsystem, QBATCH, is used, and a CL command is supplied, it becomes a message that is read by the control language processor, QCMD.

***CMD:** The input from the CMD parameter is placed in this job's message queue.

***JOBD:** The request data specified in the job description used by the job is placed in this job's message queue.

***NONE:** No request data is placed in the job's message queue.

***RTGDTA:** The routing data specified in the RTGDTA parameter of this command is placed as the last entry in the job's message queue.

'request-data': Specify the character string that is placed as the last entry in the submitted job's message queue. Up to 3000 characters can be entered, enclosed in apostrophes, if necessary. When a CL command is entered, it must be enclosed in single apostrophes, and where apostrophes would normally be used *inside* the command, double apostrophes must be used instead.

SYSLIBL

Specifies the system portion of the library being used by the submitted job.

***CURRENT:** The system portion of the library list of the submitting job is used.

***SYSVAL:** The library list specified in the system value (QSYSLIBL) at the time the job is started is used for the submitted job.

CURLIB

Specifies the name of the library being used as the current library during the processing of this command.

***CURRENT:** The current library for the job is used for the submitted job.

***USRPRF:** The current library specified in the user profile under which the submitted job runs is used. The user profile is specified on the USER parameter.

***CRTDFT:** There is no current library for the submitted job. If objects are created in the current library, the QGPL library is used as the default current library.

current-library-name: Specify the name of the library that is used as the current library of the submitted job.

INLLIBL

Specifies the user portion of the first library list that is used by the submitted job to search for any object names that are specified without a library qualifier. This does not include the system portion of the library list.

Note: Duplication of libraries in the library list is not allowed. If specifying a list of libraries for INLLIBL (or if specifying *JOBD and the job description used specifies a list of libraries), be sure that there are no duplicates contained in the list being used. To do this, compare the libraries in the list to the libraries contained in system value QSYSLIBL.

***CURRENT:** The user portion of the library list being used by the current job is used for the submitted job.

***JOBD:** The library list specified in the job description used with the job is used as the first library list for the job.

***SYSVAL:** The system default library list, QUSRLIBL, is used by the job.

***NONE:** The user portion of the first library list is empty; only the system portion is used.

library-name: Specify the names of one or more libraries that are in the user portion of the library list and are used by the submitted job. No more than 25 names can be specified. The libraries are searched in the same order as they are listed.

LOG

Specifies the message logging values used to determine the amount and type of information sent to the job log by this job. This parameter has three elements: the message (or logging) level, the message severity, and the level of message text. If no values are specified on this parameter, the values specified in the job description associated with this job are used.

Element 1: Message Level

***JOBD:** The value specified for message logging in the job description is used for the submitted job.

message-level: Specify a value, ranging from 0 through 4, that specifies the message logging level used for the submitted job's messages. For additional information on the message levels, refer to "Message Level" under the LOG parameter of the Create Job Description (CRTJOBD) command.

Element 2: Message Severity

***JOBD:** The value specified for message logging in the job description is used for the submitted job.

message-severity: Specify a value, ranging from 00 through 99, that specifies the lowest severity level to cause an error message to be logged in the job log. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Element 3: Message Text Activity

***JOBD:** The value specified for message logging in the job description is used for the submitted job.

***MSG:** Only message text is written to the job's log or shown to the user.

***SECLVL:** Second-level message text is written to the job log.

***NOLIST:** No job log is produced if the job ends normally. If the job ends abnormally (if the end-of-job code is 20 or higher), a job log is produced. The messages appearing in the job log contain message text and help text.

LOGCLPGM

Specifies whether the commands that have run in a control language program are logged to the job log by way of the CL program's message queue. This parameter sets the status of the job's logging flag. If *NO is specified, the logging flag status is off and CL commands are not logged. If *YES is specified and the LOG (*JOB) value is specified in the Create CL Program (CRTCLPGM) command, all commands in the CL program that can be logged are logged to the job log.

For more information on request logging, refer to the LOG parameter in the CRTCLPGM command description.

***JOBD:** The value specified in the job description is used.

***NO:** The commands in a CL program are not logged to the job log.

***YES:** Commands in a CL program are logged to the job log.

INQMSGRPY

Specifies the way that predefined messages that are sent as a result of running this job are answered. The user can specify that no change is made in the way that predefined messages are answered, that all inquiry messages require a reply, that a default reply be issued, or that the system reply list is checked for a matching reply as each predefined inquiry message is sent. Refer to the Add Reply List Entry (ADDRPYLE) command description for more information.

***JOBD:** The inquiry message reply control specified in the job description used with this job is used.

***RQD:** A reply is required by the receiver of the inquiry message for all inquiry messages that occur when this command is run.

***DFT:** The default reply to the inquiry message is sent. If no default reply is specified in the message description of the inquiry message, the system default reply, *N, is used.

***SYSRPYL:** The system reply list is checked to see if there is an entry for an inquiry message that is issued as a result of running this job that has a message identifier and any comparison data that match the inquiry message identifier and message data. If a match occurs, the reply value in that entry is used. If no entry exists for that message, a reply is required.

PRTDEV

Specifies the qualified name of the default printer device for this job. If OUTQ(*DEV) is specified, the file is placed on an output queue with the same name as the printer.

***CURRENT:** The printer device being used by the job that is currently running is used by the submitted job.

***USRPRF:** The printer device specified in the user profile under which the submitted job runs is used. The user profile is specified on the USER parameter.

***SYSVAL:** The value specified in the system value QPRTDEV is used.

***JOBD:** The printer device specified in the job description is used for the submitted job.

printer-device-name: Specify the name of the printer device that is used for the submitted job.

OUTQ

Specifies the qualified name of the output queue.

***CURRENT:** The output queue used by the current job is used for the submitted job.

***USRPRF:** The output queue specified in the user profile under which the submitted job runs is used. The user profile is specified on the USER parameter.

***DEV:** The output queue specified on the PRTDEV parameter is used.

***JOBD:** The output queue named in the job description used with the submitted job is the job's default output queue.

The name of the output queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

output-queue-name: Specify the qualified name of the output queue that is used as the default output queue by the submitted job.

HOLD

Specifies whether jobs using this job description are placed on the job queue in the hold condition. A job placed on the job queue in the hold condition is held until it is either released by the Release Job (RLSJOB) command or canceled by the End Job (ENDJOB) or Clear Job Queue (CLRJOBQ) command. If the job is not run before the next power-down of the system, the

job queue can be cleared (and the job ended) when the next initial program load (IPL) is done.

***JOBD:** The value specified in the job description determines whether the job is held when it is put in the job queue.

***NO:** The job is not held when it is put in the job queue.

***YES:** The spooled file is held until released by the Release Spool File (RLSSPLF) command.

SCDDATE

Specifies the date on which the submitted job becomes eligible to run.

If your system or your job is configured to use the Julian date format, the *MONTHSTR and *MONTHEND values are calculated as if the system or job did not use the Julian date format.

***CURRENT:** The submitted job becomes eligible to run on the current date.

***MONTHSTR:** The submitted job becomes eligible to run on the first day of the month. If you specify *MONTHSTR, and if today is the first day of the month, and if the time you specify on the SCDTIME parameter has not passed, the job is eligible to run today. Otherwise, the job becomes eligible to run on the first day of the next month.

***MONTHEND:** The submitted job becomes eligible to run on the last day of the month. If you specify *MONTHEND, and if today is the last day of the month, and if the time you specify on the SCDTIME parameter has not passed, the job is eligible to run today. Otherwise, the job becomes eligible to run on the last day of the next month.

***MON:** The job becomes eligible to run on Monday.

***TUE:** The job becomes eligible to run on Tuesday.

***WED:** The job becomes eligible to run on Wednesday.

***THU:** The job becomes eligible to run on Thursday.

***FRI:** The job becomes eligible to run on Friday.

***SAT:** The job becomes eligible to run on Saturday.

***SUN:** The job becomes eligible to run on Sunday.

date: Specify a date in the job date format with or without separators.

SCDTIME

Specifies the time on the scheduled date at which the job becomes eligible to run.

Note: Although the time can be specified to the second, the activity involved in submitting a job and the load on the system may affect the exact time at which the job becomes eligible to run.

The order that job entries with identical SCDDATE and SCDTIME values appear on the job queue may be different than the order in which they arrived. Likewise, these jobs may leave the job queue to be processed in an order different than the order in which they were entered. Do not assume jobs are entered or processed sequentially when they are scheduled to start at exactly the same time.

***CURRENT:** The current time is used.

time: Specify a time in the system format with or without separators defined for the job.

DATE

Specifies the date that is assigned to the submitted job when it is started.

***JOBD:** The date specified in the job description is used as the job date.

***SYSVAL:** The value in the QDATE system value at the time the job is started is used as the job date.

job-date: Specify the value that is used as the job date when the job is started. The date must be in the format specified by the system value QDATFMT. Possible formats are in the *CL Programmer's Guide*.

SWS

Specifies the first settings for a group of eight job switches that are used with the submitted job. These switches can be set or tested in a CL program and used to control the flow of the program. For example, if a certain switch is on, another program can be called. The job switches may also be valid in other high-level language programs. Only 0's (off) and 1's (on) can be specified in the 8-digit character string.

***JOBD:** The value specified in the job description is the first setting for the job's switches.

switch-settings: Specify any combination of eight 0's and 1's that is used as the first switch setting for the submitted job.

DSPSBMJOB

Specifies whether the job being submitted is allowed to be shown on the Submitted Jobs Display. Any submitted job of the type specified by the SBMFROM parameter of the Work with Submit Job (WRKSBMJOB) command can be shown if the job is not prevented by this parameter.

***YES:** This job can be shown by the WRKSBMJOB command.

***NO:** This job is not shown on any display produced by the WRKSBMJOB command.

MSGQ

Specifies the qualified name of the message queue to which messages are sent.

Note: If a job ends abnormally, the online help information of the completion message sent specifies the possible causes.

***USRPRF:** A completion message is sent to the message queue specified in the user profile of the user who submits the job.

***WRKSTN:** A completion message is sent to the work station message queue of the work station from which the job was submitted. If the job is submitted by a batch job, no completion message is sent.

***NONE:** No completion message is sent.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue to which messages are sent.

SRTSEQ

Specifies the sort sequence table to be used for string comparisons for this job.

***CURRENT:** The sort table specified for the job that is currently running is used.

***SYSVAL:** The system value QSRTSEQ is used.

***USRPRF:** The sort table specified in the user profile under which the submitted job runs is used. The user profile is specified on the USER parameter.

***HEX:** A sort sequence table is not used. The hexadecimal values of the characters are used to determine the sort sequence.

***LANGIDUNQ:** A unique-weight sort table is used.

***LANGIDSHR:** A shared-weight sort table is used.

The name of the sort sequence table can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

table-name: Specify the name of the sort sequence table to be used with this job.

LANGID

Specifies the language identifier to be used when SRTSEQ(*LANGIDUNQ) or SRTSEQ(*LANGIDSHR) is specified.

***CURRENT:** The language identifier specified for the job that is currently running is used.

***SYSVAL:** The system value QLANGID is used.

***USRPRF:** The language ID specified in the user profile under which the submitted job runs is used. The user profile is specified on the USER parameter.

language-ID: Specify the language identifier to be used by the job.

CNTRYID

Specifies the country identifier to be used by the job.

***CURRENT:** The country identifier specified for the job that is currently running is used.

***SYSVAL:** The system value QCNTRYID is used.

***USRPRF:** The country ID specified in the user profile under which the submitted job runs is used. The user profile is specified on the USER parameter.

country-ID: Specify the country identifier to be used by the job.

CCSID

Specifies the coded character set identifier (CCSID) used for the submitted job.

A CCSID is a 16-bit number identifying a specific set of encoding scheme identifiers, character set identifiers, code page identifiers, and additional coding-related information that uniquely identifies the coded graphic representation used.

***CURRENT:** The CCSID specified for the job that is currently running is used.

***USRPRF:** The CCSID specified in the user profile under which the submitted job runs is used. The user profile is specified on the USER parameter.

***SYSVAL:** The CCSID specified for the system value QCCSID at the time the job is started is used.

***HEX:** The CCSID 65535 is used.

coded-character-set-identifier: Specify the CCSID. More information on valid CCSIDs is in the *National Language Support Planning Guide*.

SBMFOR

Specifies the job name to be used on the SBMFROM parameter of the WRKSBMJOB command.

***CURRENT:** The name of the currently active job is used.

job-number/user-name/job-name: Specify the job number, user name, and job name to be used.

Note: You must have *JOBCTL authority to use this parameter.

Examples

Example 1: Submitting a Job

```
SBMJOB  JOB(SPECIAL)  JOBD(MYLIB/MYJOBD)
  CMD(CALL MYPROG)
```

This command causes the job named SPECIAL to be submitted. Most of the attributes for the job are taken from the job description MYJOBD, or the job that is currently running, except for the command. The CALL command is placed on the submitted job's message queue so that the program MYPROG can be called and run later.

Example 2: Submitting a Job

```
SBMJOB  JOB(PAYROLL)  JOBD(PAYROLL)  INQMSGRPY(*RQD)
```

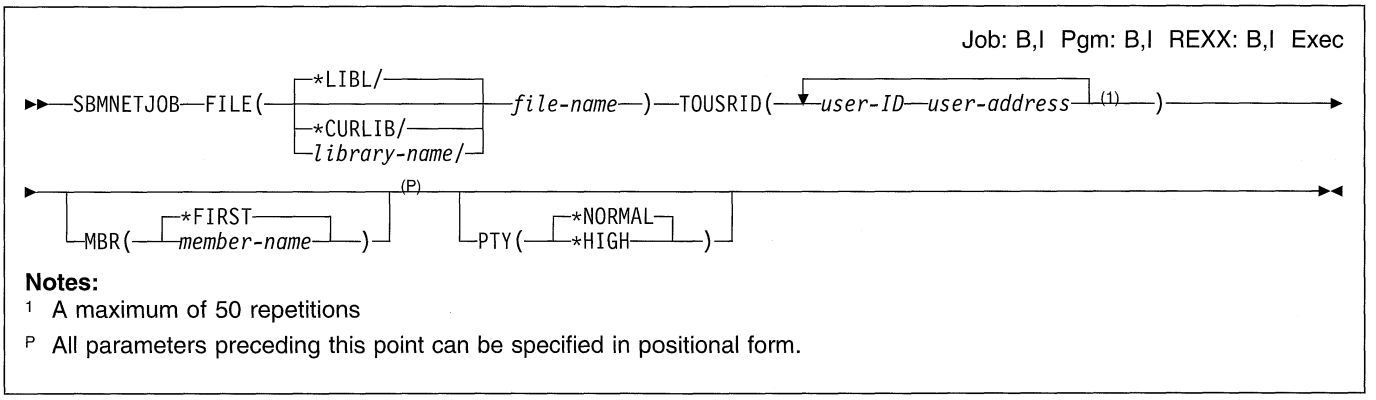
This command submits a job named PAYROLL to the system. All the information needed for this job (such as the job queue and routing data but not the inquiry message control value) is contained in the job description PAYROLL, or the job that is currently running. The library list in effect for the job issuing this command is used to find the job description. All inquiry messages sent during running of this job requires the receiver of the inquiry message to reply.

Example 3: Submitting a Job to a Job Queue

```
SBMJOB  JOBD(*USRPRF)  JOB(COPY12)  JOBQ(NIGHTQ)
  CMD(CPYF FILEA FILEB)
```

This command submits the job COPY12, which uses the job description in the user profile of the submitting job, to the job queue NIGHTQ. The CMD parameter provides the CL command necessary for the job to run. A command such as this might be used to copy the file at night while the system is unattended.

SBMNETJOB (Submit Network Job) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Submit Network Job (SBMNETJOB) command sends an input stream to another system user on the SNADS network. (The input stream is sent to another user where it can be filed, submitted, or rejected.) When the input stream arrives, its placement is governed by the job action (JOBACN) network attribute. If the value of JOBACN is *SEARCH, the entry in the network job table at the receiving system is used to determine the action taken. (The network job table entry is determined by the value specified on the ACTION parameter on the Add Network Job Entry (ADDNETJOBE) or Change Network Job Entry (CHGNETJOBE) commands.) At the receiving system, the job may be submitted immediately, filed for placement by the receiving user, or rejected.

When the input stream arrives at the destination system, a message is sent to both the recipient of the input stream as well as the originator of the input stream stating that the input stream arrived.

This command can only be used to send an input stream to a user on a remote system.

Restrictions:

1. To use this command, the user must have object operational and read authority to the file that is submitted, and for the library that contains the file.
2. The user must be enrolled in the system distribution directory to use this command. (For information on enrolling in the system distribution directory, see the *Distribution Services Network Guide*.)
3. If the job action (JOBACN) network attribute on the receiving system is set to *SEARCH, there must be an entry for the user in the network job table on the receiving system. The entry in this table specifies a user profile on the receiving system that is used to verify that the user is authorized to submit the job on that system. The user profile on the receiving system must be authorized to use the job queues, and must have object operational authority for the job descriptions specified by the JOB commands in the input stream.
4. The file that is submitted cannot contain more than approximately 2 billion bytes of data.

Required Parameter

FILE

Specifies the name of the physical file containing the input stream that is sent.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the physical file that contains the input stream being sent.

TOUSRID

Specifies the two-part user ID of one or more users to whom the input stream is submitted, or the name of one or more distribution lists containing the two-part user IDs of one or more users to whom the file is to be sent. A combination of both user IDs and distribution lists can be specified on the same command. Each user ID or distribution list is specified as a two-part name, and both parts are required.

Notes:

1. Depending on the type of work station being used, the internal value for a user identifier may differ from the characters shown by the Display Directory (DSPDIR) command. If the byte-string value specified for the TOUSRID parameter does not match the rules for an internal user identifier value, or if it does not match the internal value for any enrolled user, an error may be reported.

2. The user specified in this parameter, or in the distribution list, must be a remote user. The SBMNETJOB command cannot be used to send input streams to local users.

Optional Parameters

MBR

Specifies the member that is sent from the file.

***FIRST:** The first member in the database file is used.

member-name: Specify the name of the file member that is submitted.

PTY

Specifies the queuing priority used for the input stream when it is being routed through a SNADS network.

***NORMAL:** The input stream is sent with a service level priority of data low, which is used for most data traffic. On an AS/400 system, data low distributions are placed on the normal distribution queue specified for the route.

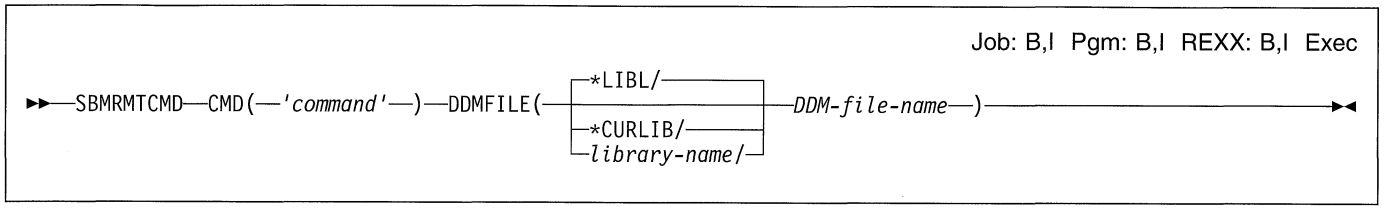
***HIGH:** The input stream is sent with a service level priority of data high, which is used for high priority data traffic. On an AS/400 system, data high distributions are placed on the data high distribution queue specified for the route.

Example

```
SBMNETJOB   FILE(PAYROLL)  TOUSRID(PAYROLL SYSTEM1)
  MBR(WEEKLY)
```

This command sends the input streams contained in member WEEKLY of file PAYROLL to user ID PAYROLL SYSTEM1.

SBMRMTCMD (Submit Remote Command) Command



Purpose

The Submit Remote Command (SBMRMTCMD) command submits a command via Distributed Data Management (DDM) for running on the target system specified by a DDM file. The value for the RMTLOCNAM parameter in the DDM file determines the communications line used, and thus indirectly identifies the target system that receives the submitted command.

This command is used to send only CL commands to another AS/400 system. It is valid only when the target system is an AS/400 system or a System/38. It can be used for AS/400 system or System/38 commands only; for example, Operation Control Language (OCL) commands cannot be sent to a target System/36.

The primary purpose of this command is to allow a source system user or program to perform file management operations and file authorization activities on files located on a target AS/400 system. More information on file management operations is in the *DDM Guide*. If the source system program or user has the correct authority, the following actions are examples of what is performed on remote files by using the SBMRMTCMD command:

- Create or delete physical, logical, device, or source files
- Grant or revoke object authority to remote files
- Check, rename, or move files or other objects
- Save or restore files or other objects

Although the command is used to do many things with files or objects, some are not as useful as others. For example, it can be used to show the file descriptions or field attributes of remote files or to dump files or other objects, but the output results remain at the target system. To display remote file descriptions and field attributes at the source system, use the DSPFD and DSPFFD commands, specifying SYSTEM(*RMT).

A secondary purpose of this command allows a user to perform nonfile operations, such as creating a message queue, or to submit user-written commands for running on the target system.

Restrictions:

1. Although remote file processing is synchronous in the user process, which includes two separate jobs (one running on each system), file processing on the target system operates independent of the source system. Commands such as OVRDBF, OVRMSGF, and DLTOVR, which are dependent on a specific recursion level or request level, may *not* function as expected.
2. Output (such as spooled files) generated by a submitted command exists only on the target system. The output is *not* sent back to the source system.
3. Some types of CL commands should *not* be submitted to a target AS/400 system. The following are examples of types that are *not* the intended purpose of the SBMRMTCMD command and that may produce undesirable results.
   - All OVRxxxF commands that refer to device files, communications files, message files, or save files.
   - The DSPxxxx commands should not be submitted because the output results remain at the target system.
   - Commands that are valid only in an interactive environment, like WRKxxx or STRxxx.
   - Job-related commands like RRTJOB that are used to control a target system's job. The CHGJOB command, however, *can* be used.
   - Commands that are used to service programs, like SRVJOB, TRCJOB, TRCINT, or DMPJOB.
   - Commands that may cause inquiry messages to be sent to the system operator, like STRPRTWTR or CPYTODKT. Pass-through can be used instead.
4. Translation is not performed for any *impromptu* messages caused by target system errors, because they are not stored on the system; the text for an impromptu message is sent directly to the source system shown. The message identifier of all other message types generated on the remote system is sent back to the source system. If the message text that exists for the message identifier on the source system has been translated, it will then be shown.
5. Up to 10 messages, generated during the running of a submitted command, can be sent by the target system to the source system. If more than 10 messages are generated, an additional *informational* message is sent that indicates that the messages exist in the job log for the target job on the target system. If one of those messages is an *escape* message, the first nine messages of other types are sent, followed by the informational message and the escape message.

Required Parameters

CMD

Specifies a character string of up to 2000 characters that represents a command that is run on the target system.

The command must be allowed in both *BATCH and *EXEC (QCAEXEC) environments on the target AS/400 system.

The command must be enclosed in apostrophes if it contains embedded blanks or special characters.

Note: The normal rule of pairing apostrophes in quoted strings on the local system must be *doubled* when the same string is submitted to a remote system on this CMD parameter; this is required because the user is coding a quoted string within another quoted string. Therefore, when this parameter is being coded, wherever a single apostrophe would normally be paired with another apostrophe, *each occurrence in the outside set of apostrophes must be doubled* to produce the same results at the target system.

DDMFILE

Specifies the qualified name of the DDM file that is used to submit the command to the target system. If no library qualifier is given, *LIBL is used to find the file. The DDM file is used only to determine the remote location representing the target system. The remote file name associated with the DDM file is ignored by the SBMRMTCMD command.

The name of the DDM file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

DDM-file-name: Specify the name of the DDM file used to submit the command to the target system.

Examples

Example 1: Deleting a File

```
SBMRMTCMD  CMD('DLTF LIBX/FRED')  DDMFILE(DENVER)
```

This command deletes the file named FRED in library LIBX on the target system that is associated with the DDM file named DENVER.

Example 2: Creating a Physical File

```
SBMRMTCMD  CMD('CRTPF  SALES/CAR  SRCFILE(QGPL/QDDSSRC)
  SRCMBR(MASTER)')  DDMFILE(DENVER)
```

This command creates the physical file CAR in library SALES using the data description specifications (DDS) in the source file QDDSSRC and source member named MASTER in the QGPL library. The DDS must already exist on the target system identified by the DDM file named DENVER in the target job's library list.

Example 3: Changing the Text Description

```
SBMRMTCMD  CMD('CHGDDMF  FILE(LIBX/STANLEY)
  TEXT(''Don''''t forget to pair apostrophes.'')')
  DDMFILE(SMITH)
```

This command changes the text in the description of the DDM file named STANLEY which is stored in library LIBX. Because the submitted command requires an outside set of single apostrophes (for the CMD parameter), each single or double apostrophe normally required in the TEXT parameter for *local* system processing must be doubled for *remote* system processing. The coding above produces a single apostrophe in the text when it is shown or printed on the remote system.

Example 4: Creating a DDM File

```
SBMRMTCMD CMD('CRTDDMF FILE(SALES/MONTHLY)
  RMTFILE(*NONSTD ''CAR.SALES(JULY)''
  RMTLOCNAME(DALLAS)') DDMFILE(CHICAGO)
```

This command creates (on the target system identified by the information in the DDM file named CHICAGO) another DDM file named MONTHLY. The new DDM file is stored in a library named SALES on the CHICAGO system. The new DDM file on the CHICAGO system is used to access a file and *member* on a different system named DALLAS. The accessed file is named SALES/CAR and the member name in the file is JULY.

Note that this CRTDDMF command string contains *three* sets of single apostrophes: one set to enclose the entire command being submitted, and a double set to enclose the file and member named in the RMTFILE parameter. This is how any IBM AS/400 system file *member* name must be specified on the SBMRMTCMD command, because of the parentheses needed to enclose the member name.

Example 5: Replacing a Portion of the Library List

```
SBMRMTCMD  CMD('RPLLIBL  LIBL(QGPL QTEMP SALES EVANS)')
  DDMFILE(EVANS)
```

This command replaces the user's portion of the library list being used by the target job associated with the DDM file named EVANS, which is being used by the source job in which this SBMRMTCMD command is being submitted. In that source job, if there are other open DDM files that specify the same device and mode, this library list is used for them also.

Additional Considerations

Override example

The DDMFILE parameter is used to determine the target system to which the command is sent. Overrides that apply to the DDM file (not the remote file) are taken into account for this function. For example, if a file override was in effect for a DDM file because of the following commands, which override FILEA with FILEX, then the target system to which

the DLTF command is sent is the one associated with the remote location values specified in FILEX (the values point to the DENVER system, in this case).

```
CRTDDMF   FILE(SRCLIB/FILEA)   RMTFILE(SALES/CAR)
  RMTLOCNAME(CHICAGO)

CRTDDMF   FILE(SRCLIB/FILEX)   RMTFILE(SALES/CAR)
  RMTLOCNAME(DENVER)

OVRDBF    FILE(FILEA)   TOFILE(SRCLIB/FILEX)
  SBMRMTCMD   CMD('DLTF RMTLIB/FRED')
  DDMFILE(SRCLIB/FILEA)
```

Distributed Data Management conversations: When a SBMRMTCMD command runs on the target system, it has a target system job associated with it. Successive SBMRMTCMD commands that are submitted using the same DDM file and DDM conversation may run in the same or different target system jobs, depending on the DDMCNV job attribute value. The DDMCNV job attribute value determines whether the DDM conversation is dropped (DDMCNV(*DROP)) or remains active (DDMCNV(*KEEP)) when the submitted function has completed. If the conversation is dropped, the next SBMRMTCMD command runs using a different target job.

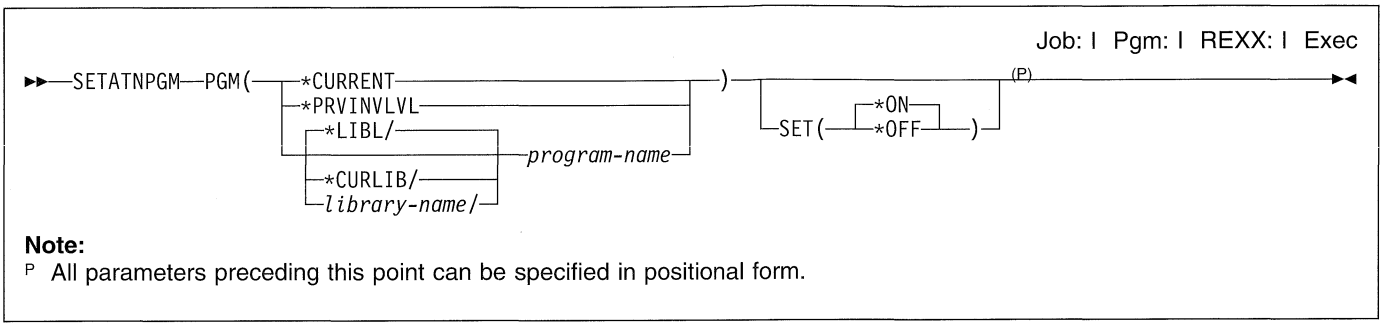
Command syntax checking: The syntax of the command character string being submitted by the CMD parameter is not checked by the source system. For example, in the case of a user-defined command, the command definition object may not exist on the source system.

Command running results: Because the submitted command runs as part of the target system's job, the attributes of that job (such as the library search list, user profile, wait times, and running priority) may cause a different result than if the command were run locally.

Error message handling:

- For errors detected by the target system when processing the command that was submitted, the source system attempts to send to the source system user the same error information that the target system sent. However, if the source system does not have an equivalent message for the one sent by the target system, source system default message text (stating that the message does not exist on the source system) is sent to the source system user, along with the message identifier and message type of the actual target system message. The target system messages can be viewed on the source system by using pass-through and either the DSPJOB or DSPJOBLOG command.
- If the SBMRMTCMD command is used to call a CL program on the target system, unmonitored escape messages generated by the program are changed into inquiry messages and sent to the system operator. If the user does not want the target system operator to have to respond to this inquiry message before the job can continue, the user can refer to the following commands and do either of the following on the target system:
  - If the user wants to specify a default reply for a specific job, the INQMSGRPY parameter on either the CRTJOBD or CHGJOBD command can be used to specify either *DFT or *SYSRPYL in the job description for the target job.
  - If the user wants to specify a default reply message for a specific inquiry message, the ADDRPYLE command can be used (on the target system) to add an entry to the system-wide automatic message reply list for that message.

SETATNPGM (Set Attention Program) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Set Attention Program (SETATNPGM) command causes the specified program to become the attention key handler at the current recursion level in the job running the command. If the ATTN key handler's status is SET(*ON), the specified program is called when the ATTN key is pressed. No parameters are passed to the ATTN key handler when it is called. The attention handling program runs in the same process with the same job attributes, overrides, and group authorities as the program that issued the SETATNPGM command. However, program adopted authority is not carried over.

This command is call-oriented; that is, a SETATNPGM command issued at one recursion level causes the attention handling program to be in effect at the current recursion level as well as deeper recursion levels, until another SETATNPGM command occurs which changes the attention handling program. More information is in the *Work Management Guide*.

Pressing the ATTN key under the following conditions does not call the ATTN key handler (that is, it appears as if SET(*OFF) were the status of the ATTN key):

- When the input inhibited light at the work station is on
- At the system request menu or any of its options
- At the break message display and the display message display
- While the OS/400 system is already calling the attention handler
- Some OS/400 system functions (3270 Emulation, 5250 Passthru) do not allow the ATTN key to call the attention handling program

Note: Caution is necessary when defining an attention handling program. Attention handling is similar to system request option 1 (transfer to secondary job), but there are special considerations for its use. When a transfer to secondary job occurs, processing continues in another job; however, when the ATTN key is pressed, the ATTN key handler runs in the same job. The program in progress when the ATTN key is pressed is not protected by any locks held. This means that the attention handler can get a lock on any object, even if a program it interrupted already had an exclusive lock. Be aware that read from invited devices operation could time-out during the time that the attention key handling program is running. Therefore, if a time-out were to complete in the program in progress while the attention key handling program is running, whatever action taken as a result of that time-out occurs upon return to the program in progress.

For example, if the WAITRCD value in the file is set to 60 seconds, the program is set to exit if a key is not pressed in 1 minute, and the attention key program is invoked and runs longer than 1 minute, the program exits when it returns from the attention key handler.

However, caution should be used because a check for data available is made before checking to see that the time-out completes. If a key is pressed immediately after leaving the attention key handler, data could be available which could complete the read from invited devices and the time-out is not checked. This could cause unexpected results.

Applications that perform Get-No-Wait operations can be interrupted at any point where the ATTN key is pressed. Attention handlers therefore should:

- Use simple functions like a menu which transfers to a group job or secondary job.
- Avoid referring to objects or functions that may be in use when the ATTN key is pressed.
- Avoid calling nonrecursive functions (many non-OS/400 system functions) that are going to be interrupted by the ATTN key. (This is especially true of high-level languages and utilities, because they are not designed to be used recursively.)

Restriction: To use this command the program specified must exist and you must have object operational authority or one of the data authorities to it. You must also have read authority to the program's library.

Required Parameters

PGM

Specifies the qualified name of the program to be the ATTN key handler at this recursion level.

***CURRENT:** The program name of the ATTN key handler currently in effect is used as the value of this parameter. An error does not occur if an ATTN key handler is not currently in effect.

***PRVINVLVL:** The ATTN key handler in effect at the previous recursion level is reinstated as the ATTN key handler at this recursion level. The SET parameter is not allowed if this special value is specified, because the SET status of the previous recursion level is also reinstated. This option is used when a program has specified an attention program and wants to revert back to a previous level.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of the program that is the ATTN key handler.

Optional Parameters

SET

Specifies whether the ATTN key handler indicated in the program (PGM) parameter should be called when the ATTN key is pressed. This parameter is not allowed if PGM(*PRVINVLVL) is specified.

***ON:** The ATTN key handler specified in the program (PGM) parameter is called when the ATTN key is pressed.

***OFF:** The ATTN key handler specified in the program (PGM) parameter is not called when the ATTN key is pressed.

Examples

Example 1: Setting the ATTN Key Handler

```
SETATNPGM  PGM(QGPL/ATTN)  SET(*ON)
```

This command causes the program QGPL/ATTN to become the ATTN key handler. Because SET(*ON) is specified, the program is called when the ATTN key is pressed.

Example 2: Setting the Attention Key Off

```
SETATNPGM  PGM(*CURRENT)  SET(*OFF)
```

The current attention handling program has its status changed to SET(*OFF). Because the status is SET(*OFF) when the ATTN key is pressed, the attention handling program is not called.

Example 3: Previous Recursion-Level Support

```
SETATNPGM  PGM(*PRVINVLVL)
```

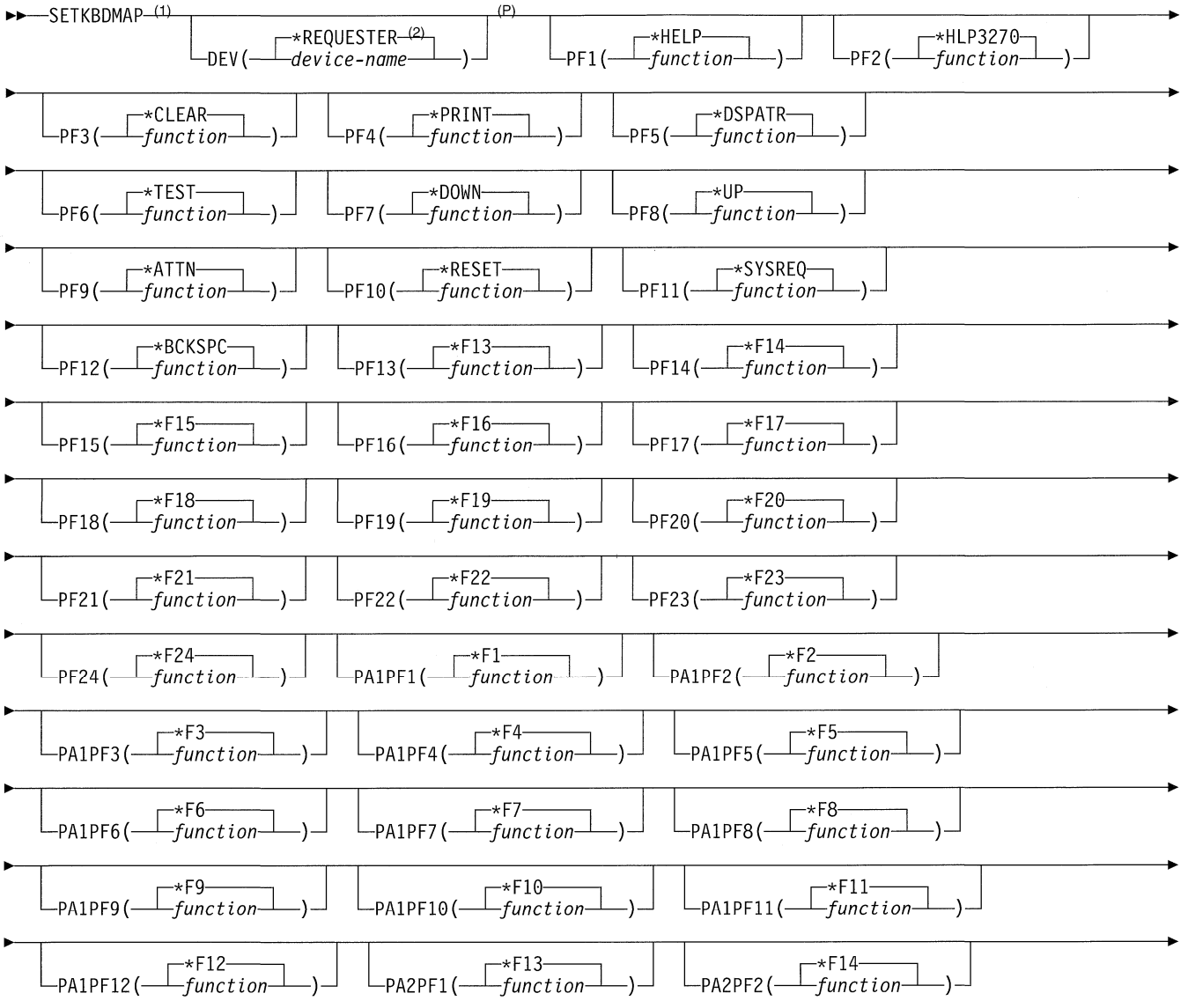
The attention handling program and status that was in effect at the previous recursion level is reinstated at this recursion level. If no attention handler is in effect, after this command is run nothing happens when the ATTN key is pressed.

Example 4: Emulating the System Request Key

```
SETATNPGM  PGM(QWSSYSRQ)
```

The system-supplied program QWSSYSRQ will be called when the ATTN key is pressed. This system program allows the ATTN key to act as a system request key by showing the system request menu on the display when the ATTN key is pressed.

SETKBDMAP (Set Keyboard Map) Command

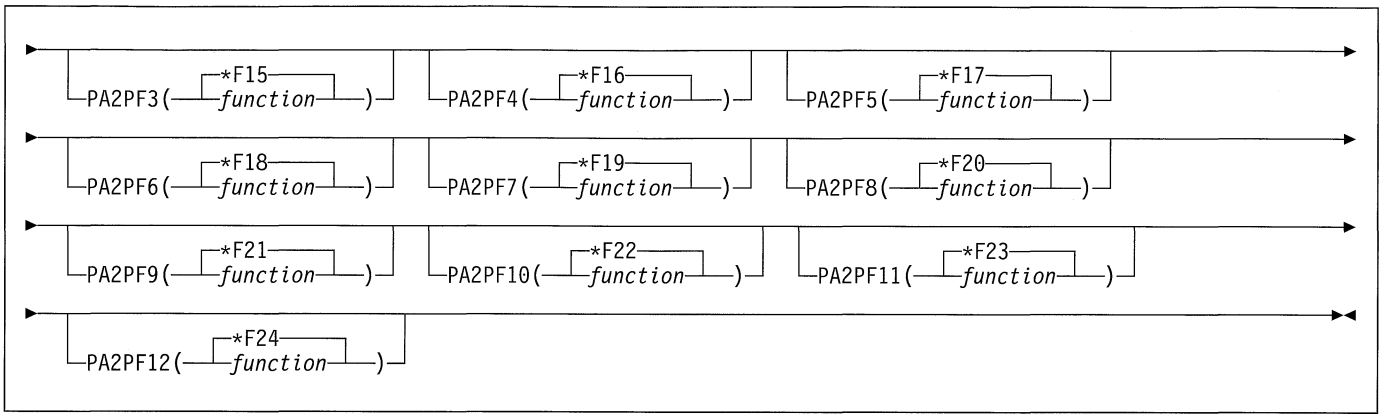


Notes:

1 The following are allowable special values which are valid for all parameters: *HELP, *HLP3270, *PRINT, *DSPATR, *TEST, *DOWN, *UP, *NONE, *RESET, *SYSREQ, *BCKSPC, *ATTN, and *F1 through *F24.

2 *REQUESTER does not work in batch.

P All parameters preceding this point can be specified in positional form.



Purpose

The Set Keyboard Map (SETKBDMAP) command allows the user to override the PA (Program Attention) and PF (Program Function) key assignment defaults provided by the AS/400 system.

This command assigns the specified F-to-PF map to the device where the command was entered (if it is a 3270 display station device) or to the 3270 display station specified if the user has authority to that device. More information on the user-assignment keyboard mapping is in the *New User's Guide*.

Optional Parameters

DEV

Specifies a valid 3270 display station that is assigned this keyboard mapping function.

***REQUESTER:** This mapping is assigned to the device where the command is entered.

device-name: Specify a device other than the device named in the *REQUESTER value. The user must have allocation authority to the specified device. In a program environment, the user should either acquire or allocate the specified device before entering this command.

Key Sequence

Each valid key or key sequence is a separate parameter which, except for restrictions noted below, may be assigned any function. If the user does not specify a function for a particular key or key sequence, the default value becomes the function for that key or key sequence. Following is a list of functions and restrictions:

*ATTN	Attention
*BCKSPC	Record Backspace
*CLEAR	Clear Screen
*DOWN	Roll Down
*DSPATR	Display Imbedded Attributes
*F1–*F24	F1 through F24 Function Keys
*HELP	5250 Help
*HLP3270	3270 Help Text (Display Active Keyboard Map)
*NONE	No Assignment
*PRINT	Print Screen
*RESET	Error reset
*SAME	No Change
*SYSREQ	System Request
*TEST	Test Request
*UP	Roll up

Notes:

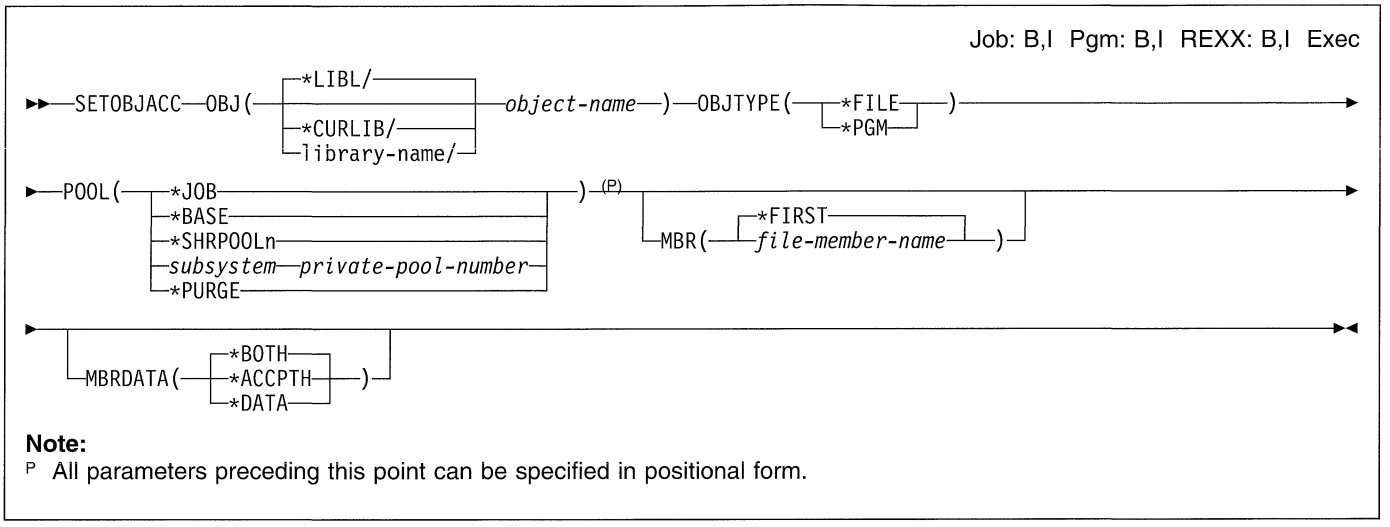
1. *HELP, *HLP3270, and *RESET must be assigned to PF1 through PF12, PA1/PF1 through PF12, or PA2/PF1 through PF12. Because these functions are required for 3270 display station device support to function properly, and not all keyboards have 24 PF keys, *HELP, *HLP3270, or *RESET may not be assigned to keys PF13 through PF24, unless that function is also assigned to PF1 through PF12, PA1/PF1 through PF12, or PA2/PF1 through PF12.
2. It is recommended that *F1 and *SYSREQ be assigned to PF1 through PF12, PA1/PF1 through PF12, or PA2/PF1 through PF12.
3. The value *ATTN cannot be explicitly assigned to a 3270 remote attach display station. If the default value *ATTN is taken, the value *NONE is substituted. However, if the value *ATTN is explicitly chosen, a diagnostic message is sent.

Example

```
SETKBDMAP   PF1(*F1)  PF2(*F2)  PF3(*F3)
  PF4(*F4)  PF5(*HLP3270)  PF9(*HELP)
```

This command reassigns the keyboard primarily for an application that makes frequent use of the 5250 CF keys F1, F2, F3, F4. All other PF key sequences are set to the default shown on the command prompt. The above command is started in the program that started the application (thus tailoring the display station to whatever application is run).

SETOBJACC (Set Object Access) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Set Object Access (SETOBJACC) command temporarily changes the speed of access to an object by bringing the object into a main storage pool or purging it from all main storage pools. An object can be kept resident in main storage by selecting a pool for the object that has available space and does not have jobs associated with it. Repeated use of the command can cause a set of objects to be resident in a main storage pool.

Required Parameters

OBJ

Specifies the qualified name of the object to be brought into or purged from main storage.

The name of the specified object can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

object-name: Specify the name of the object.

OBJTYPE

Specifies the type of object to be brought into or purged from main storage.

***FILE:** The object is a file.

***PGM:** The object is a program.

POOL

Specifies whether the object is brought into or purged from main storage.

***JOB:** The object is brought into the pool associated with the job.

***BASE:** The object is brought into the base pool.

***SHRPOOLn:** The object is brought into a general-purpose shared pool. Valid pool values range from 1 through 10.

Element 1: Subsystem

subsystem: Specify a subsystem name.

Element 2: Pool Identifier

pool-identifier: Specify a subsystem pool identifier.

***PURGE:** The object is purged from all pools.

Optional Parameters

MBR

Specifies the database file member to be brought into or purged from main storage.

***FIRST:** The first member is selected.

file-member-name: Specify the member name.

MBRDATA

Specifies the member data to be brought into or purged from main storage.

***BOTH:** All parts of the object are selected.

***ACCPTH:** The file member's access path is selected.

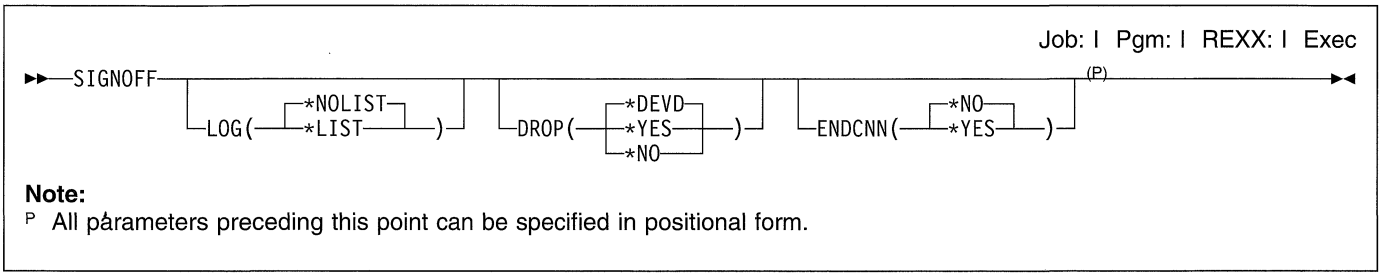
***DATA:** The file member's data is selected.

Example

```
SETOBJACC  OBJ(OBJA)  OBJTYPE(*PGM)  POOL(*JOB)
```

This command brings a program named OBJA to the pool associated with the job in which the command was executed.

SIGNOFF (Sign Off) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Sign Off (SIGNOFF) command ends an interactive job or causes all group jobs on a work station to end. The user enters this command to sign off at a work station.

Restriction: This command is valid only in an interactive job.

Optional Parameters

LOG

Specifies whether the job log for this interactive job is deleted or is included in the job's spooled output for printing. This entry takes precedence over the LOG parameter value specified for the job.

***NOLIST:** The information in the job log is deleted when the job ends.

***LIST:** The job log is spooled for printing along with the rest of the job's spooled output.

DROP

Specifies, for switched lines only, whether the switched line attached to the work station is disconnected (dropped) if no other work stations on the same line are signed on. This parameter is ignored if the work station is attached to a nonswitched line.

***DEVD:** The value specified in the DROP parameter of the work station's device description is assumed.

***YES:** The switched line is disconnected when the job is ended, if no other work stations are signed on the line.

***NO:** The switched line is not disconnected when the job is ended.

ENDCNN

Specifies whether to end the connection to the current system. Ending the connection allows the user to bypass the sign-on display of the target system and return to the source system. For communication functions that do not support this option, this parameter is ignored.

***NO:** The connection does not end. The sign-on display of the target system is shown.

***YES:** The connection ends and the user is returned to the source system. No sign-on screen or error messages are shown from the target system.

Examples

Example 1: Signing Off and Ending an Interactive Job

```
SIGNOFF
```

This command signs off the user of the work station and ends the interactive job. The switched line is dropped only if specified in the device description of this work station and if no other work station on this line is active. An end-of-job message that gives the job start and stop times is written in the job's log.

Example 2: Printing the Job Log

```
SIGNOFF  LOG(*LIST)  DROP(*NO)
```

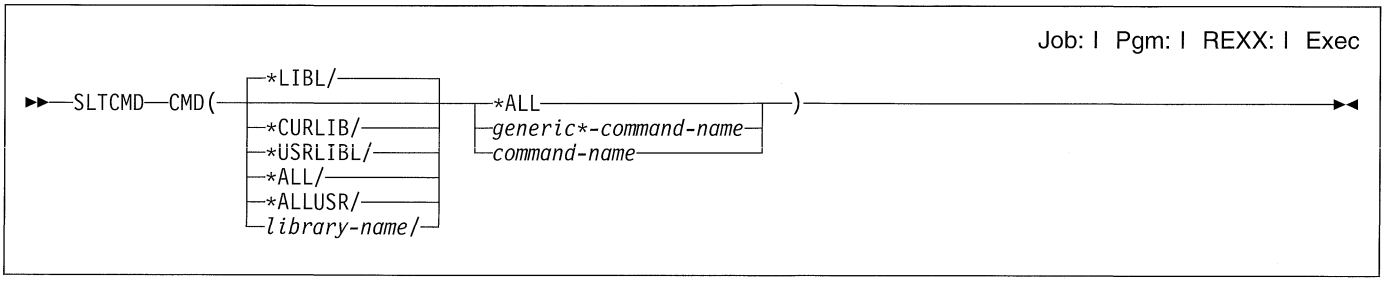
This command ends the interactive job, but the switched line is not released. The job log is printed with the job's spooled output.

Example 3: Signing Off and Ending the Connection

```
SIGNOFF  ENDCNN(*YES)
```

This command ends the connection and transfers the user back to the source system.

SLTCMD (Select Command) Command



Purpose

The Select Command (SLTCMD) command produces a list of commands from a user-specified subset of command names. From this list, only the option to select the command is available.

Restrictions: (1) Only the libraries to which the user has USE authority will be searched. (2) Only the commands to which the user has some authority will be shown on the display. (3) To perform operations on the commands, the user must have USE authority to the command used by the operation, and the appropriate authority to the commands on which the operation is to be performed.

Required Parameters

CMD

Specifies the qualified name of the commands being shown on the Select Command display.

The name of the command can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All commands in the libraries specified are listed on the Select Command display.

generic-command-name:* Specify the generic name of the command being listed. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be selected only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

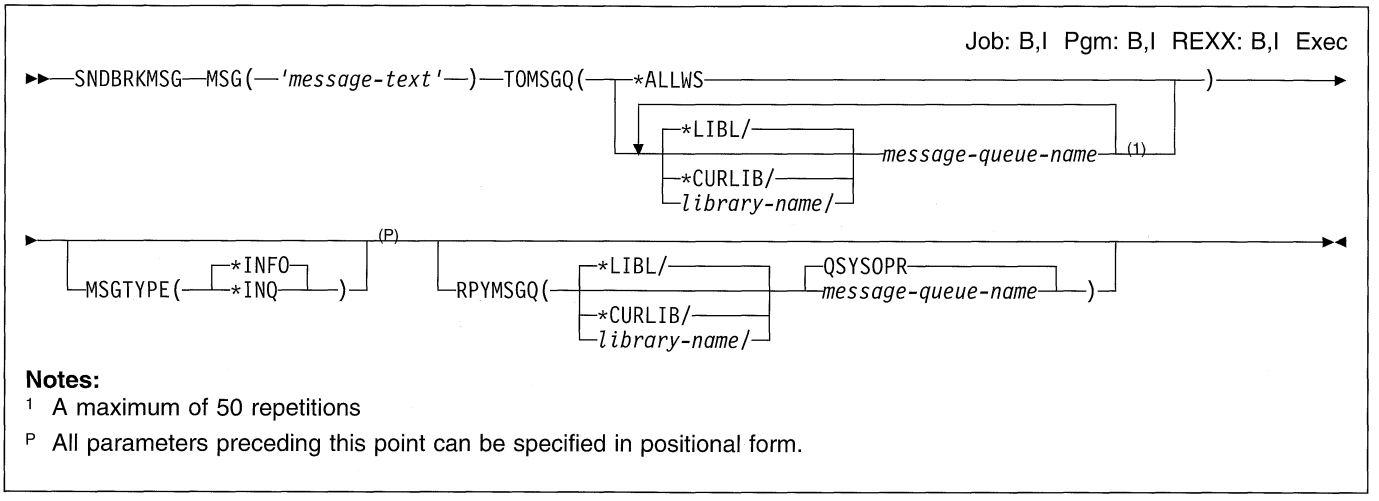
command-name: Specify the name of the command being listed.

Example

```
SLTCMD  CMD(QSYS/*ALL)
```

This command shows a list of all commands in library QSYS. The option to prompt and run commands is available.

SNDBRKMSG (Send Break Message) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Send Break Message (SNDBRKMSG) command is used by the system operator to send an impromptu message to one or more work station message queues. An impromptu message is a message that is not predefined and is not stored in a message file. The command causes the message to be delivered always in break mode. The DSPMSG display is shown for the message when it is received, regardless of the setting of the message queue's delivery mode, severity, and break handling program. However, the message may not be displayed in some cases, depending on the BRKMSG job attribute. This command is primarily intended for the system operator's use.

Restrictions:

1. This command can be used to send break messages to work station message queues only.
2. This command cannot send inquiry messages (specified by MSGTYPE(INQ)) to multiple work stations.

Required Parameters

MSG

Specifies the message text that is sent. A maximum of 512 characters can be specified. The message must be enclosed in apostrophes if it contains blanks or other special characters.

TOMSGQ

Specifies the qualified names of one or more work station message queues to which the break message is sent. Only the names of work station message queues can be specified.

***ALLWS:** The break message is sent to all work stations and Personal Computer message queues. This value cannot be specified if *INQ is specified on the MSGTYPE parameter.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue to which the break message is being sent.

Optional Parameters

MSGTYPE

Specifies the type of message that is sent in break mode. Only informational or inquiry message types can be specified.

***INFO:** An information-only message is sent in break mode.

***INQ:** An inquiry message is sent in break mode; the work station receiving the message is expected to reply to it. Inquiry messages can be sent to only one work station at a time.

RPYMSGQ

Specifies, only if an inquiry message is sent, the name of the message queue to which the work station user's reply is sent.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QSYSOPR: The replies to the break message are sent to the system operator's message queue, QSYSOPR.

message-queue-name: Specify the name of the message queue to which a reply to the break message is sent. Only a user or work station message queue can be specified.

Examples

Example 1: Sending a Message

```
SNDBRKMSG   MSG('The inventory application
  shuts down at 4:00 pm today.')
```

This command sends the message 'The inventory application shuts down at 4:00 pm today.' to all work station message queues. If the work station is signed on, the message will be delivered in break mode regardless of the delivery attribute setting of those message queues. The message is also added to the work station message queues of those work stations that are not signed on.

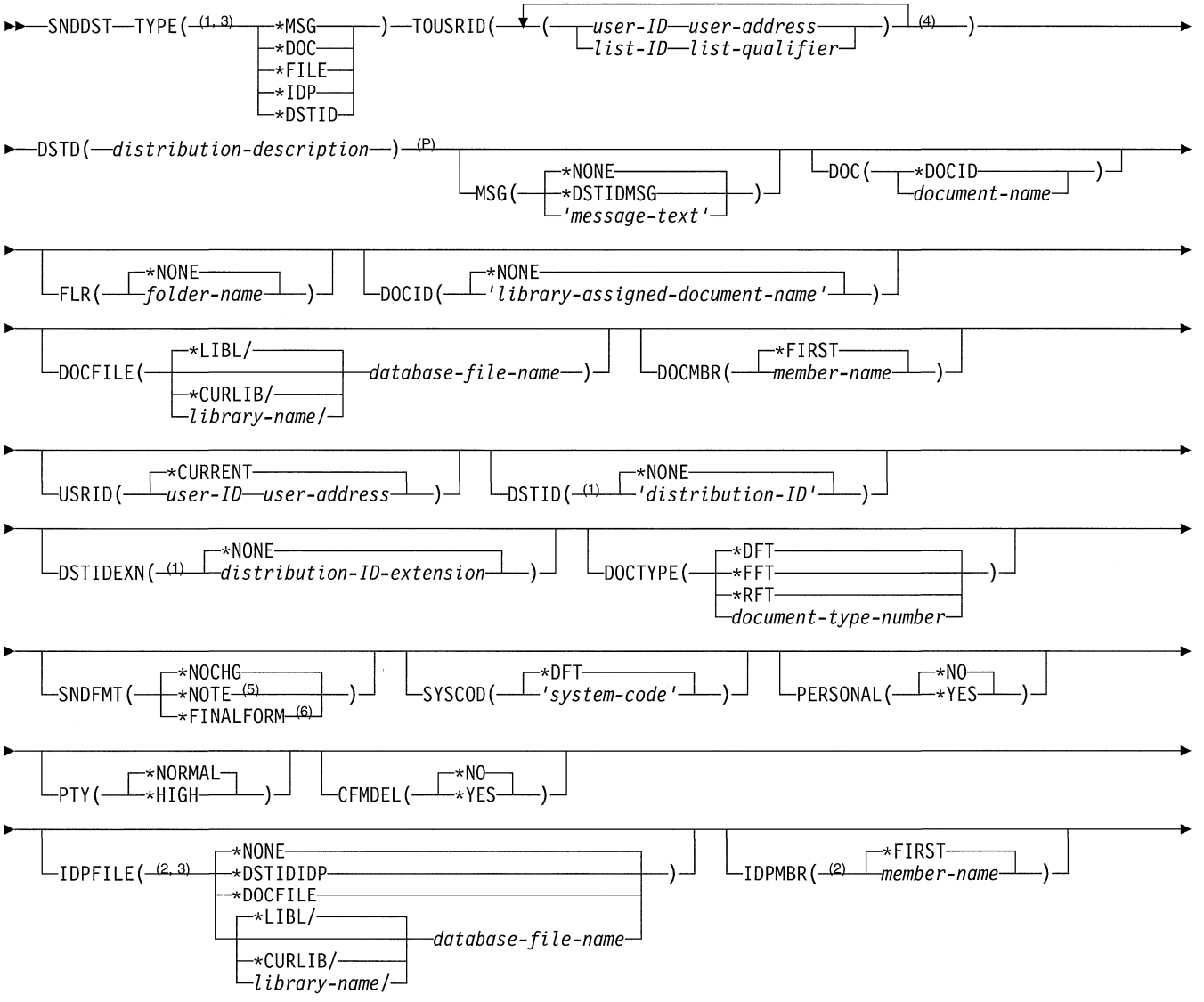
Example 2: Sending an Impromptu Message

```
SNDBRKMSG   MSG('Your printed output is ready.')
  TOMSGQ(GEORGEMSGQ)
```

This example shows a typical use of the SNDBRKMSG command by the system operator to send an impromptu message to a work station user.

SNDDST (Send Distribution) Command

Job: B,I Pgm: B,I REXX: B,I Exec



Notes:

1 The DSTID and DSTIDEXN parameters are valid only when TYPE(*DSTID) is specified.

2 The IDPMBR parameter is only used when data base-file-name is specified on the IDPFILE parameter.

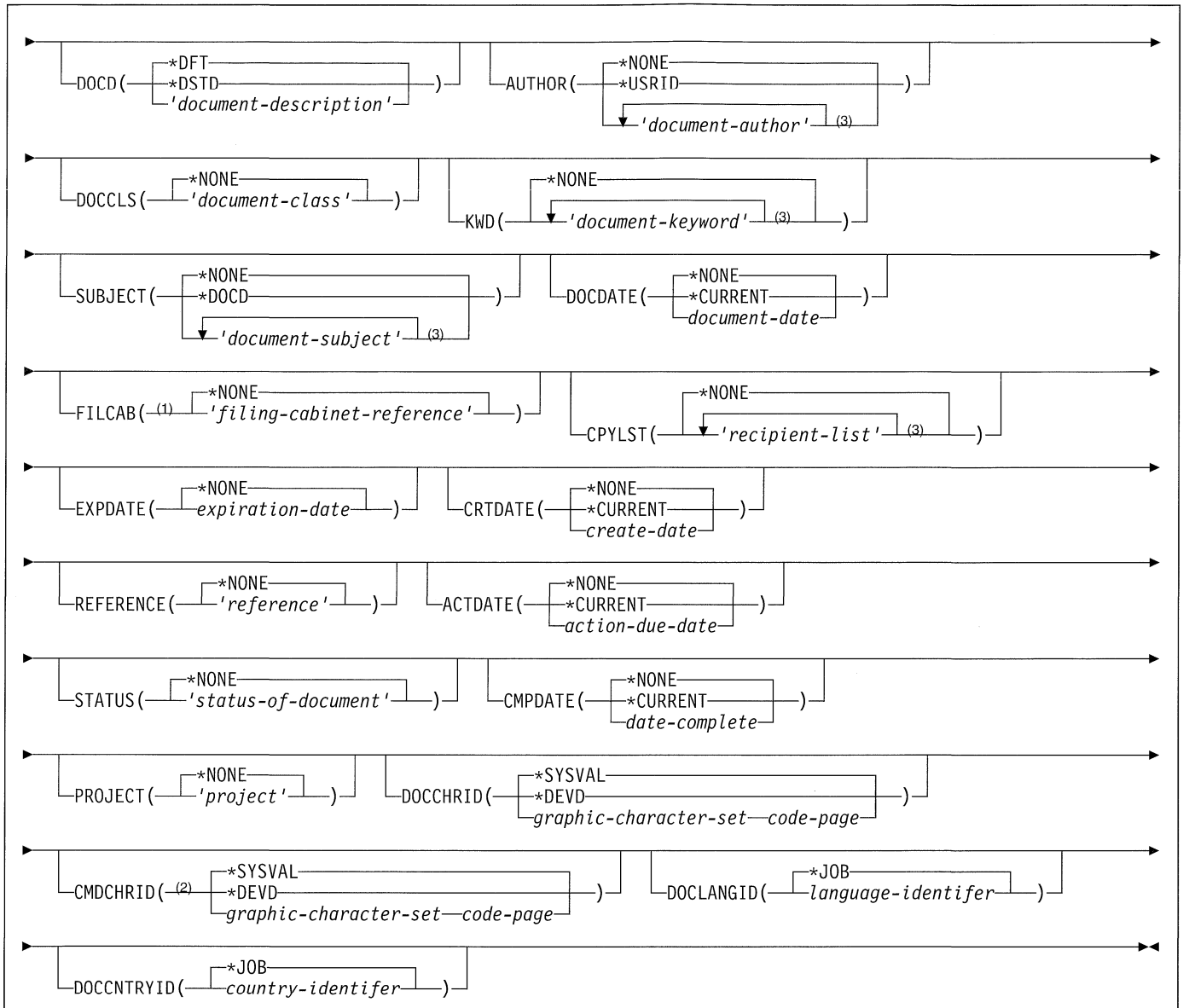
3 FILCAB is required when TYPE(*IDP) and IDPFILE(*NONE) are specified.

4 A maximum of 300 repetitions

P All parameters preceding this point can be specified in positional form.

5 SNDFMT(*NOTE) is valid only when *DOC, *FILE, or *DSTID is specified on the TYPE parameter.

6 SNDFMT(*FINALFORM) is valid only when TYPE(*DOC) is specified.



Notes:

1 FILCAB is required when TYPE(*IDP) and IDPFILE(*NONE) are specified.

2 This parameter applies to the following parameters: TOUSRID, USRID, SYSCOD, MSG, DOCD, DSTD, AUTHOR, DOCCLS, KWD, FILCAB, CPYLST, SUBJECT, REFERENCE, STATUS, PROJECT, and DSTID.

3 A maximum of 50 repetitions

Purpose

The Send Distribution (SNDDST) command allows a user to send distribution to another user, to a list of users, or to a distribution list.

Restrictions:

1. A current user of this command working on behalf of another user must be granted authority to work on the behalf of the other user by means of the Grant User Permission (GRTUSRPMN) command.
2. Personal distribution cannot be received by a requester working on behalf of another user.
3. The DOCD, SUBJECT, and DSTD parameters are not updated from the SNDDST command to the actual mail log when a document (*DOC is specified for the TYPE parameter) is sent.

Required Parameters

TYPE

Specifies the type of information sent and the valid values on this command. Dependent parameter codes for this parameter (see syntax diagram) determine which parameters are valid. If the dependent parameter code

is listed in the syntax diagram, the parameter is valid when that type is specified.

***MSG:** Only the message specified in the MSG parameter is sent.

***DOC:** The document specified in the DOC or DOCID parameter is sent. The user must have authority for the document before it can be sent.

***FILE:** The database file specified in the DOCFILE and DOCMBR parameters is sent. The database file is sent without any changes. The user must have authority for the database file before it can be sent.

***IDP:** The interchange document profile (IDP) specified in the IDPFILE and the IDPMBR parameters, or the document profile built from the keywords DOCD, AUTHOR, DOCCLS, KWD, SUBJECT, REFERENCE, STATUS, PROJECT, ACTDATE, CMPDATE, CRTDATE, DOCDATE, FILCAB, CPYLST, and/or EXPDATE is used. The profile can be a profile of a printed document.

***DSTID:** The mail entry identified by the distribution ID is distributed. The distribution ID is referred to as the distribution document name.

TOUSRID

Specifies the user ID and address of one or more users to whom the distribution is being sent, or the distribution list name of one or more distribution lists containing the user ID and address of one or more users to whom the distribution is being sent. A combination of user IDs and distribution lists can be used on the same command. Up to 300 user IDs and addresses can be specified.

DSTD

Specifies the description of the distribution. Specify the description of the distribution. Up to 44 characters can be specified.

Optional Parameters

MSG

Specifies a short message.

***NONE:** No message is sent with the distribution.

***DSTIDMSG:** The same message in the distribution document specified on the DSTID parameter is sent with the distribution.

'message-text': Specify the message to send to the users. Up to 256 characters enclosed in apostrophes can be specified.

DOC

Specifies the name of the document being sent.

***DOCID:** The document being sent is identified by the library-assigned document name.

document-name: Specify the user-assigned name of the document being sent. Up to 12 characters can be specified.

FLR

Specifies the name of the folder that contains the document. This is the user-assigned name given to the folder when it is created. If the DOC parameter is specified, the FLR parameter must also be specified.

***NONE:** No folder is specified when the document is identified by the DOCID parameter.

folder-name: Specify the name of the folder that contains the document being sent. A folder name can consist of a series of folder names (FLRA/FLRB/and so on) if the document being sent is located in a folder contained in another folder. Up to 63 characters can be specified.

DOCID

Specifies the library-assigned name of the document being sent. This is the name assigned to the document by the system when it was created. The library-assigned document names can be determined by using the Query Document Library (QRYDOCLIB) command or the library-assigned document name is returned in a completion code when using the File Document (FILDOC) command. Library-assigned document names are 24 characters in length with the following format: YYYYMMDDHHMNSSHSSNSNSNSN where:

YYYY = year

MM = month

DD = day

HH = hour

MN = minute

SS = second

HS = hundredth of a second

SNSNSNSN = system name

***NONE:** No library-assigned document name is required when the document is identified by the DOC parameter.

'library-assigned-document-name': Specify the library-assigned document name being sent.

DOCFILE

Specifies the name of the database file that contains the document data to be sent. The database file is a user-defined file, or it is the output file specified by RCVDST(OUTFILE) or RTVDOC(OUTFILE). If an output file is specified, only the data portion of the document data record is read from the output file and the prefix is removed from the document data record. More information on defining the format of database files (OUTFILES) is in the *Database Guide*.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file.

DOCMBR

Specifies the document database file member to be sent.

***FIRST:** The first member in creation order in the database file is sent.

member-name: Specify the name of the database file member to be sent.

USRID

Specifies the user ID and address of the user for whom the request is made.

***CURRENT:** The user profile under which the current job is running is used.

Element 1: User ID

user-ID: Specify the user ID of the user to whom the distribution is sent.

Element 2: User Address

user-address: Specify the user address of the user to whom the distribution is sent.

DSTID

Specifies the unique distribution ID of the distribution. The ID is assigned to the distribution by the system that originated it.

Distribution IDs can be found by using the Query Distribution (QRYDST) command. IDs are also returned from the Send Distribution (SNDDST) command.

***NONE:** No distribution ID is sent.

'distribution-ID': Specify a distribution ID composed of the second part of the sender's user ID (padded on the right with blanks to 8 characters), the first part of the sender's user ID (padded on the right with blanks to 8 characters), and a 4-digit zoned sequence number with leading zeros. For example, *NEWYORK SMITH 0204*. The distribution ID is entered this way because a blank character is valid with a user ID address. This parameter is required when TYPE(*DSTID) is specified.

DSTIDEXN

Specifies the extension of the distribution ID if one is specified on the DSTID parameter. This extension uniquely identifies a duplicate distribution. This extension is a 2-digit extension ranging from 00 through 99. For incoming distributions, this extension ranges from 01 through 99. For confirmation of delivery distributions, this extension must be 00. For example, if the distribution ID is *NEWYORK SMITH 0204* and two copies of this distribution were sent to a user, then the user has two distributions with the same distribution ID. To uniquely distinguish the two distributions, an extension is added to each distribution. For example, one distribution is identified by distribution ID and extension *NEWYORK SMITH 020401* and the other one by *NEWYORK SMITH 020402*. If there are no duplicate IDs, the extension defaults to 01. These extensions map one-to-one with the distribution ID that is specified on the DSTID parameter.

***NONE:** There is no duplicate distribution. This value is equivalent to a 01 extension.

distribution-ID-extension: Specify the extension associated with the distribution. This is used to uniquely identify duplicate distributions.

DOCTYPE

Specifies the type of document being distributed. This identifier is used by the system to determine whether it can correctly handle the data stream.

***DFT:** The system creates the right document type identifier based upon the source of the data.

Document Type	Description
2	Final form text document
3	5520 revisable form text document
4	Word processing EBCDIC
5	Word processing information file
6	Image-data subset document
7	3730 text data stream
8	DIA document library descriptor document
9	3732 display document data stream
10	DIA display document data stream
11	Revisable form text document
12	1403 compatible data stream with variable length, unblocked records
13	Digitized ADS audio
14	IBM PC data file
15	Hard copy of document
223	Data file

If TYPE(*IDP) is specified, the type is 15 (printed document).

If TYPE(*FILE) is specified and the file is an output file created by the RTVDOC command (OUTFILE parameter), then the document type is taken from this file. If this file does not contain a document type or the value in the file for the document type is 0, an error message is sent.

If the file is a user-defined file, the document type is 223 (data file).

If TYPE(*DOC) is specified, the document type of the document sent is used. If TYPE(*DSTID) is specified, the document type is the same document type that is associated with the distribution.

***FFT:** The document is a final-form document. This type of document is intended to be shown and printed, but not edited, by the receiver.

***RFT:** The document is a revisable-form text. This type of document can be viewed, printed, and edited by the receiver.

document-type-number: Specify a value ranging from 2 through 65535. The numbers ranging from 2 through 32767 are controlled by registering them with the IBM Document Interchange Architecture and are used for IBM-defined document types. The numbers ranging from 32768 through 65535 are not registered with IBM and can be used for non-IBM defined document types. The meaning of these document types must be stated as values of the SYSCOD parameter.

SNDFMT

Allows the user to specify the format of the document being sent.

***NOCHG:** The document is sent in the current format.

***NOTE:** The document is sent in a final form text document content architecture (FFTDCA) data stream as a note.

FINALFORM: The document is sent in FFTDCA.

SYSCOD

Specifies text used with the document type parameter to help identify the type of document being sent. The receiver of the data stream determines the document data stream and processing requirements to edit, view, print or change the document.

***DFT:** The system supplies a default system code. If the DOCTYPE value is within the range 2 through 32767, the default is 'IBM AS/400 CL'. If the DOCTYPE is within the range 32768 through 65535, a system code must be specified.

'system-code': Specify the text that helps uniquely identify the type of document being sent. Up to 13 characters can be specified.

PERSONAL

Specifies whether the distribution is sent as a personal distribution. A user working on behalf of another user cannot receive this distribution.

***NO:** The distribution is not sent as a personal distribution.

***YES:** The distribution is sent as a personal distribution.

PTY

Specifies whether the distribution is sent using normal priority or high priority. For distributions to remote receivers, the priority determines which 'SNADS Next System Queue' is used. Normal priority distributions use the normal next system queue. High priority uses the priority next system queue. The difference between high and normal priority depends on information, specified by the user, on the Configuration Distribution Services (CFGDSTSRV) command. The handling of priority distributions by other office system nodes can vary, but generally the high priority distributions take the faster path when there is a choice of paths. For distribution to local receivers, the priority determines whether a message is sent to the receiver's message queue to notify the receiver of the distribution. No message is sent for normal distributions.

***NORMAL:** Normal priority is used.

***HIGH:** High priority is used.

CFMDEL

Specifies whether the sender wants a confirmation of delivery notification when each receiver receives the distribution. If confirmation of delivery is not requested, the sender is still informed if the distribution is not delivered because of system failures, routing failures, or because it has an invalid user ID. If confirmation of delivery is requested, the sender is informed when the receiver either receives the distribution or deletes the distribution before receiving it. To get this information, the sender must do a Query Distribution (QRYDST) command with the OPTION(*OUT) specified.

***NO:** Request for confirmation of delivery is not specified.

***YES:** Request for confirmation of delivery is specified.

IDPFILE

Specifies the name of a database file that contains the interchange document profile (IDP) data. The interchange document profile is either a user-defined file or the output file specified on the OUTFILE parameter of the Receive Distribution (RCVDST) or Retrieve Document (RTVDOC) commands. If an output file is specified, only the data portion of the document profile record is read from the output file and the prefix is removed from the IDP data.

If this parameter is specified, the remaining parameters on this command are ignored except for the DOCCHRID and CMDCHRID parameters.

***NONE:** The interchange document profile is not in a database file. The information is provided by other parameters on this command.

***DSTIDIDP:** The interchange document profile (IDP) information associated with the distribution document is used. This is valid only when TYPE(*DSTID) is specified.

***DOCFILE:** The interchange document profile is in the same file as the document. A member specified in the IDPMBR parameter is ignored.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file that contains the interchange document profile. More information on defining the format of database files (output files) is in the *Database Guide*.

IDPMBR

Specifies the interchange document file member name to send. This parameter is valid only when IDPFILE(database-file-name) is specified.

***FIRST:** The first member (based on creation date) in the database file is sent.

member-name: Specify the name of the database file member to be sent.

DOCD

Specifies a description for the document being distributed. This is the Document Interchange Architecture IDP document name field.

***DFT:** The system creates a document description for database files. The default is library-name/file-name/member-name. For a description of a printed document (TYPE(*IDP)), the default is the distribution description. For a description of a distribution document (TYPE(*DSTID)), the default is the document description associated with the distribution.

***DSTD:** The distribution description is used for the document name.

'document-description': Specify up to 44 characters for the description of the document.

AUTHOR

Specifies the authors of the document.

***NONE:** No author is identified for the document.

***USRID:** The user ID and address of the user who is requesting the distribution is used as the author's name.

'document-author': Specify the name of the authors. Up to 50 authors can be specified.

DOCCLS

Specifies the class associated with this document, such as MEMO, FORM, or SHEET.

***NONE:** No class is assigned to the document.

'document-class': Specify the document class. Up to 16 characters can be specified.

KWD

Specifies the keywords that can be used to describe the document.

***NONE:** No keywords are defined for this document.

'document-keyword': Specify the keywords to use to describe the document. Up to 50 keywords can be specified and each keyword can have up to 60 characters.

SUBJECT

Specify the subjects of the document.

***NONE:** No subjects are defined for the document.

***DOCD:** The document description is used as the subject for the document.

'document-subject': Specify the subjects of the document. Up to 50 subjects can be specified and each subject can have up to 60 characters of text.

DOCDATE

Specifies the date the user wants to assign to the document.

***NONE:** No date is assigned to the document.

***CURRENT:** The current date is used.

document-date: Specify the value used as the document date. The date must be in the format specified by the system value QDATFMT.

FILCAB

Specifies the location of the document. This parameter is intended for printed documents. The interchange document profile, which refers to the printed document, is distributed. This parameter is required if TYPE(*IDP) and IDPFILE(*NONE) are specified.

***NONE:** No filing cabinet reference is defined for this document.

'filing-cabinet-reference': Specify the text that describes where the printed document is located. Up to 60 characters can be specified.

CPYLST

Specifies names of the users who receive this distribution.

***NONE:** No copy list is included for this distribution.

'recipient-list': Specify the names of the users who receive the document. Up to 50 names can be specified and each name can have up to 60 characters.

EXPDATE

Specifies the date on which the document is no longer needed.

***NONE:** No expiration date is specified.

expiration-date: Specify the value to use as the expiration date. The date must be specified in the format specified by the system value QDATFMT.

CRTDATE

Specifies the creation date of the object.

***NONE:** A document creation date is not specified.

***CURRENT:** The current date is used.

create-date: Specify the document creation date. The date must be in the format specified by the system value QDATFMT.

REFERENCE

Specifies a reference associated with the document.

***NONE:** No reference field is included in the interchange document profile.

'reference': Specify the text that describes the reference associated with the document. Up to 60 characters can be used.

ACTDATE

Specifies the due date for the requested action.

***NONE:** No action due date is specified.

***CURRENT:** The current date is used.

action-due-date: Specify the value used as the action due date. The date must be specified in the format specified by the system value QDATFMT.

STATUS

Specifies the user-defined status (In Process, Pending Approval, or Retired).

***NONE:** No status is included in the interchange document profile.

'status-of-document': Specify text that describes the status of the document. Up to 20 characters can be specified.

CMPDATE

Specifies the completion date for the requested action.

***NONE:** No completion date is specified.

***CURRENT:** The current date is used.

date-complete: Specify the completion date. The date must be in the format specified by the system value QDATFMT.

PROJECT

Specifies the project with which the document is associated.

***NONE:** No project field is included in the interchange document profile.

'project': Specify text that describes the project of the document. Up to 10 characters can be specified.

DOCCHRID

Specifies the character identifier (graphic character set and code page) for the data being distributed. The character identifier is related to the display device that was used to create the data being distributed. More information about CHRID processing is in the *Database Guide*.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values to use for creating the data being distributed.

Element 2: Code Page

code-page: Specify the code page value used to create the command parameters. Valid values range from 1 through 999.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the *Guide to Programming Displays*.

Note: The CMDCHRID parameter applies to the following parameters and means in some cases that the data is translated to a code page and character set that is interchangeable with other IBM office products. The interchangeable character set and code page is '697 500' except for USRID, TOUSRID, and DSTID which are '930 500'. In other cases, the code page and character set are attached to the field and sent with it to allow the receiving terminal to correctly print and display the field.

Parameter	Translate
TOUSRID	Translated
USRID	Translated
DSTID	Translated
SYSCOD	Translated
MSG	Translated
DOCD	Passed
DSTD	Translated
AUTHOR	Passed
DOCCLS	Passed
KWD	Passed
SUBJECT	Passed
FILCAB	Passed
CPYLST	Passed
REFERENCE	Passed
STATUS	Passed
PROJECT	Passed

The value translates the USRID and DSTID parameter to character set and code page of '930 500'. The *Distribution Services Network Guide* contains the character set and code page table for '930 500'.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from

an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values to use for creating the command parameter.

Element 2: Code Page

code-page: Specify the code page. Valid values range from 1 through 9999.

DOCLANGID

Specifies the language identifier being placed in this distribution document's interchange document profile (IDP). This parameter is ignored if the IDPFILE parameter is specified.

***JOB:** The language identifier specified for the job in which this command is entered is used.

language-identifier: Specify the language identifier. More information on valid language identifiers is in the *National Language Support Planning Guide*. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

DOCCNTRYID

Specifies the country identifier being placed in this distribution document's interchange document profile (IDP). This parameter is ignored if the IDPFILE parameter is specified.

***JOB:** The country identifier specified for the job in which this command is entered is used.

country-identifier: Specify an ISO 3166 Alpha-2 code from the country code table. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Examples

Example 1: Sending a Distribution on Behalf of Another User

```
SNDDST  TYPE(*FILE)  TOUSRID((JACKSON RCH38DB))
  DOCTYPE(20000)  SYSCOD(BRANDX)
  DOCFILE(DEPT46ELIB/XTEXT)  DOCMBR(GOLD1IPFS)
  PTY(*HIGH) USRID(JACOBSON RCH38NBS)
  DSTD('IPFS FOR GOLD1 PROJECT')
  MSG('review and update section 1.2.4.
  return for final printing by 2/15/89')
  CFMDEL(*YES)
```

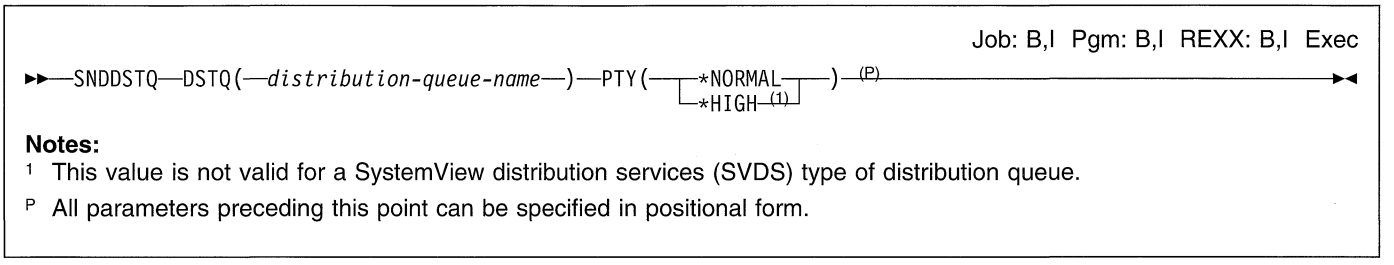
This command sends a distribution that is being sent by someone (such as a secretary) who is authorized to work on behalf of JACOBSON. The document being sent is a BRANDX text document that is sent to another user who also has the BRANDX text processor.

Example 2: Sending a Mail Log Entry

```
SNDDST  TYPE(*DSTID)  DSTID('NEWYORK SMITH 0204')
  DSTIDEXN(02)  TOUSRID((JACKSON RCH38DB))
  MSG(*DSTIDMSG)  CFMDEL(*YES)
```

This command sends a mail log entry that is identified by the distribution document name *NEWYORK SMITH 0204* that is distributed to user JACKSON at address RCH38DB. The message in the distribution document is distributed with the distribution.

SNDDSTQ (Send Distribution Queue) Command



Notes:

1 This value is not valid for a SystemView distribution services (SVDS) type of distribution queue.

P All parameters preceding this point can be specified in positional form.

Purpose

The Send Distribution Queue (SNDDSTQ) command is used:

- To send a distribution queue's entries when the distribution queue is configured to be manually started but no operator is available.
- To override any distribution queue scheduling attributes and begin sending a queue's entries immediately.
- To restart a SNADS sender job that failed abnormally.

The SNDDSTQ command is primarily intended for use in a batch CL program. The SNDDSTQ command enables the same functions as option 2 (Send distribution queue) on the Work with Distribution Queue (WRKDSTQ) command main list panel. The SNDDSTQ command allows the functions to be started from a batch job instead of interactively.

Distribution queue names are translated to the graphic character set and code page 930 500, using the job's coded character set identifier (CCSID).

Restrictions:

1. This command is shipped with public *EXCLUDE authority, and the QPGMR and QSYSOPR user profiles have private authorities to use the command.
2. Messages that report errors about distribution queues may display or print different characters than the user entered for the distribution queue name because of internal system transformations. Similarly (depending on the language used for the work station), the internal value for a distribution queue name may differ from the characters shown on the Work with Distribution Queue (WRKDSTQ) command. An error may be reported if the character-string value specified for the DSTQ parameter does not match the rules for an internal distribution queue value or if it does not match the internal value for any defined distribution queue (ignoring case differences).

Required Parameters

DSTQ

Specifies the name of the distribution queue being sent. Specify the name of the distribution queue. The queue specified must have been previously configured using the Configure Distribution Services (CFGDSTSRV) command or the Add Distribution Queue (ADDDSTQ) command.

PTY

Specifies whether the normal or high priority portion of the specified queue is sent.

***NORMAL:** Sends the normal priority queue, which is for distributions with a service level of data low.

***HIGH:** Sends the high priority queue, which is for distributions with a service level of fast, status, or data high.

Examples

Example 1: Sending Distributions with Normal Priority

```
SNDDSTQ  DSTQ(CHICAGO)  PTY(*NORMAL)
```

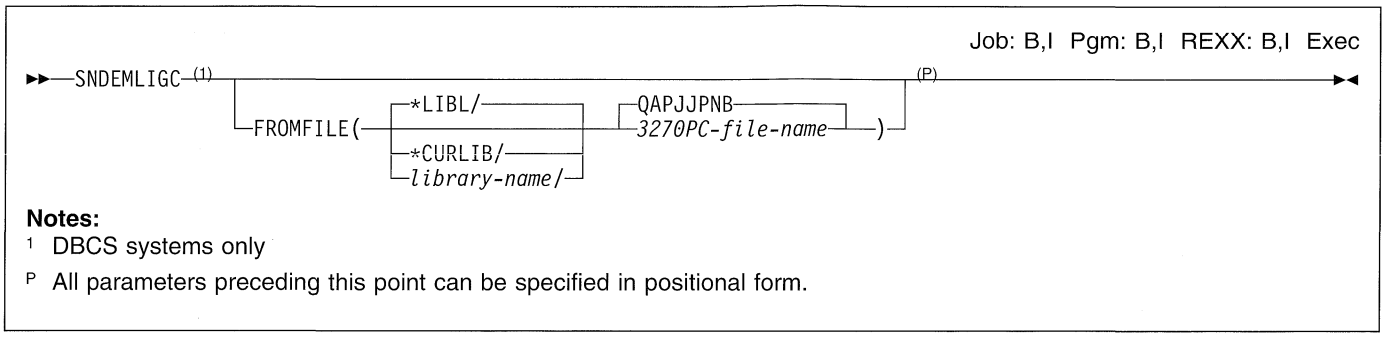
This command sends distributions from the normal priority portion of the CHICAGO distribution queue.

Example 2: Sending Distributions with High Priority

```
SNDDSTQ  DSTQ(ATLANTA)  PTY(*HIGH)
```

This command sends distributions from the high priority portion of the ATLANTA distribution queue.

SNDEMLIGC (Send DBCS 3270PC Emulation Code) Command



Notes:
1 DBCS systems only
P All parameters preceding this point can be specified in positional form.

Purpose

The Send Double-Byte Character Set (DBCS) 3270PC Emulation Code (SNDEMLIGC) command sends a list of instructions to the system to download Personal System/55 (PS/55) Personal Computer code from an AS/400 system to a PS/55 Personal Computer. Downloading is required if the user receives message CPF8557 while running the Start 3270 Device Emulation command (STREML3270) with *YES specified for the IGCEMLPC parameter.

The downloaded code is used to do SNA DBCS 3270PC emulation using the PS/55 Personal Computer. Emulation of SNA DBCS 3270PC display stations and printers connected to a Systems Network Architecture (SNA) 3274 is done from the downloaded code. SNA DBCS 3270PC emulation is called by the Start 3270 Display Emulation (STREML3270) command. The STREML3270 command must specify IGCEMLPC(*YES) requesting SNA DBCS 3270PC emulation. Additional information on SNA DBCS 3270PC emulation is in the *SNA DBCS 3270PC Emulation User's Guide*, associated with the national language configured for your PS/55.

Optional Parameters

FROMFILE

Specifies the qualified name of the 3270PC file being downloaded to the PS/55.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QAPJJPNB: The 3270PC file is downloaded to a Japanese PS/55 with an 8086 or 80286 processor.

Other IBM-supplied 3270 PC files that can be downloaded are:

QAPJJPNE: For a Japanese PS/55 with an 80386 processor.

QAPJJPNV: For a Japanese PS/55 with DOS J5.0.

QAPJKORB: For a Korean PS/55 with an 8086 or 80286 processor.

QAPJKORE: For a Korean PS/55 with an 80386 processor.

QAPJTCHB: For a Traditional Chinese PS/55 with an 8086 or 80286 processor.

QAPJTCHE: For a Traditional Chinese PS/55 with an 80386 processor.

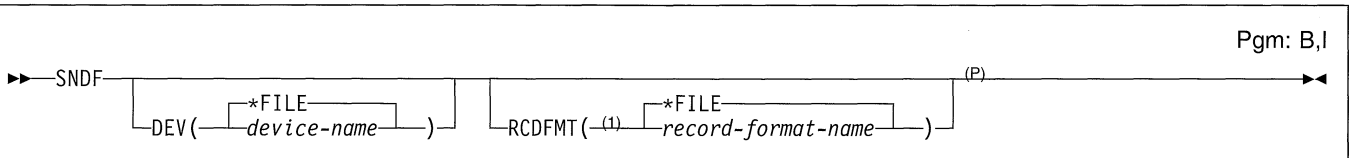
3270PC-file-name: Specify the name of the 3270 PC file to be downloaded to your personal computer.

Example

```
SNDEMLIGC  FROMFILE(QAPJJPNE)
```

This command informs the user of the steps to be taken to download the 3270PC file QAPJJPNE to a Japanese PS/55 with an 80386 processor.

SNDF (Send File) Command



Notes:
1 A CL variable cannot be coded on this parameter.
P All parameters preceding this point can be specified in positional form.

Purpose

The Send File (SNDF) command is used by a control language (CL) program to send a record to a display device that is being used by an interactive user. The device can be any display station including the console. The command sends the data from the program's CL variables to the display's device file in the specified record format. These variables were automatically declared in the program (one for each field in the record format) when the CL source program was compiled and a Declare File (DCLF) command was processed as part of the source.

Of the record formats specified in the DCLF command, only one can be specified in each SNDF command. If the device file has not been opened, it is opened by this command. The file and record format specified in this command can be overridden by an Override with Display File (OVRDSPF) command if it is entered before the file is opened. However, care should be taken that the fields in the overriding record format correspond to the CL variables declared in the program.

Restrictions: (1) This command is valid only within a CL program. (2) This command is valid only for display files. (3) It cannot be used with database files.

Optional Parameters

DEV

Specifies the name of the display device to which the data in the CL variables for the specified record format is sent.

***FILE:** The program's data is sent to the device associated with the device file that was declared in the FILE parameter of the DCLF command. If more than one device name is specified in the device file, *FILE cannot be specified.

device-name: Specify the name of the device or the name of the CL variable that contains the name of the device to which the program's data is sent.

RCDFMT

Specifies the name of the record format that is used to send data to the file. The format contains all the fields in the record. This parameter must be coded with a record format name if there is more than one record format name in the device file; *FILE cannot be coded if there is more than one. If the record format contains the INVITE DDS keyword (optioned on), the SNDF functions as if SNDRCVF WAIT(*NO) had been coded.

***FILE:** There is only one record format in the device file; that is the format in which the program's data is sent to the file.

record-format-name: Specify the name of the record format in which the program's data is sent to the file. A CL variable cannot be used here to specify the record format name.

Example

```
DCLF   FILE(MENU1)
  •
  •
  •
SNDF
```

The record format in the device file MENU1 is sent to the device specified in the file. There is only one record format in the file.

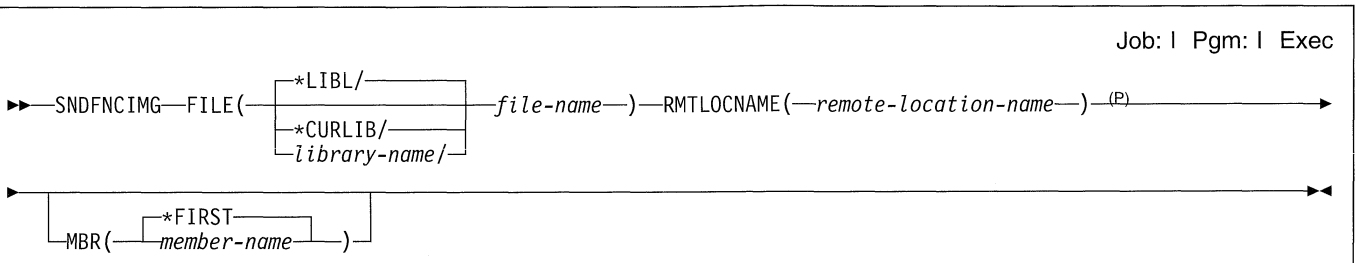
```
DCLF   FILE(SCREEN1)  RCDFMT(REC1 REC2)
  •
  •
  •
SNDF   DEV(DISP3)  RCDFMT(REC1)
```

The device file named SCREEN1 causes the display station named DISP3 to display the data sent by the CL program to the user. The data is shown in the format specified by the REC1 record format.

```
DCLF   SCREEN1  (REC1 REC2)
  •
  •
  •
SNDF   *N  REC2
```

The device file named SCREEN1 sends data to the device named in the same device file. *N is the null value for the DEV parameter, coded positionally, that defaults to *FILE. The data is presented to the user in the format specified by REC2.

SNDFNCIMG (Send Finance Diskette Image) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Send Finance Diskette Image (SNDFNCIMG) command allows the user to send an operational diskette image to a 4701 finance controller, which takes the image and builds a new operational diskette that is used to start the controller.

The user must first create the diskette image on a host system (for example, System/370*) by using the Host Diskette Image Create package. The user must also provide a host program to block the image into a basic exchange file format and then provide a way of sending the blocked image to an AS/400 system and placing it in a file. This command rebuilds the original image and sends it to the controller, which writes the image on a blank diskette and builds an operating diskette containing system code, configuration files, and application programs. The diskette can then be used to start the controller.

Restriction: The user must have QSECOFR authority to use this command.

Required Parameters

FILE

Specifies the qualified file name where the diskette image is located.

The file being sent must be a physical file with a record length of 80 bytes. Save files, logical files and device files are not allowed. Overrides to the specified file are ignored.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the file where the diskette image is located.

RMTLOCNAME

Specifies the remote location name of a finance device with TYPE(*FNCICF). This device must be attached to a 4701 finance controller with an 8-inch diskette drive or a 3601 controller configured as a 4701 controller. The local location address (LOCADR) of the device *must* be 01.

Optional Parameters

MBR

Specifies the member in the file that contains the diskette image that is blocked into a basic exchange format.

***FIRST:** The first member in the database file is used.

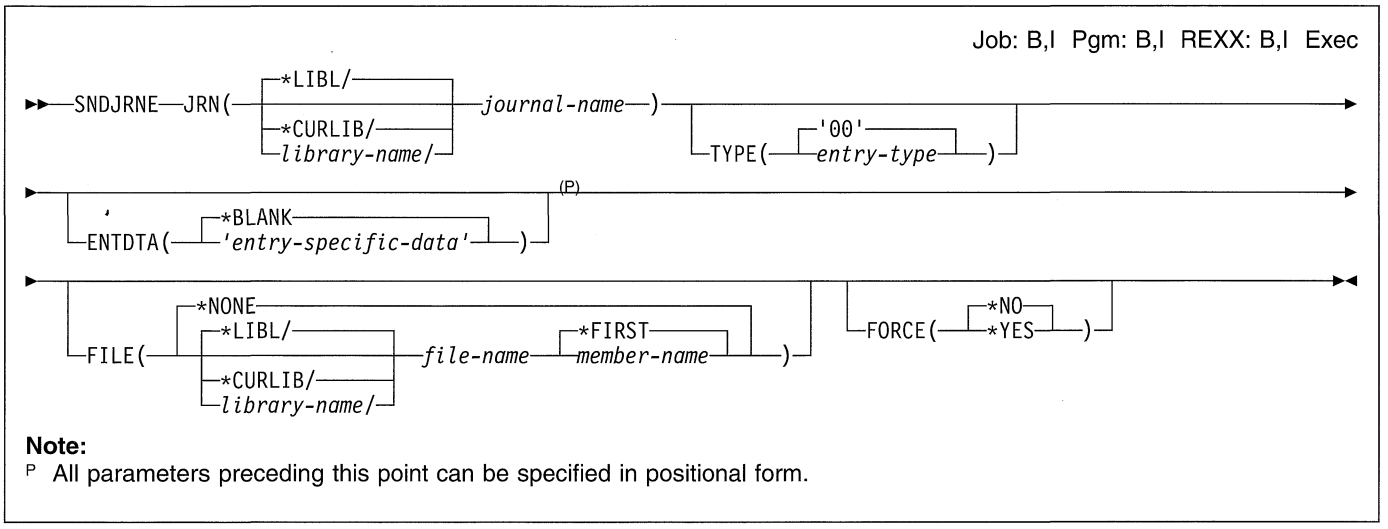
member-name: Specify the name of the member that contains the image to be blocked.

Example

```
SNDFNCIMG  FILE(IMAGEFILE)  MBR(OTSIMAGE)
  RMTLOCNAME(SYSMON1)
```

This command builds the diskette image from member OTSIMAGE in file IMAGEFILE and then sends it to remote location SYSMON1.

SNDJRNE (Send Journal Entry) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Send Journal Entry (SNDJRNE) command is used to write a single journal entry to a specific journal. The entry can contain any information. The user may assign an entry type to the journal entry and may also associate the journal entry with a specified physical file member.

Note: The journal code for the entry is 'U', which indicates a user-specified journal entry.

Restriction: If a file is specified, its entries must either be currently journaled to the specified journal or they must have been last journaled to the specified journal.

Required Parameter

JRN

Specifies the qualified name of the journal that contains the new journal entry.

The name of the journal can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

journal-name: Specify the name of the journal that contains the new journal entry.

Optional Parameters

TYPE

Specifies the journal entry type of this journal entry.

'00': The journal entry type is a '00' (hex F0F0).

entry-type: Specify a 2-character value or hexadecimal value used for the journal entry type. This value must be greater than or equal to hex C000.

If a hexadecimal value is specified that does not represent characters, that value is not shown on the DSPJRN display or on the printout.

ENTDTA

Specifies the user-specified data that is placed in the variable portion of the journal entry.

***BLANK:** Text is not specified.

'entry-specific-data': Specify up to 3000 characters, enclosed in apostrophes.

FILE

Specifies the qualified name of the database physical file and member with which this entry is associated.

Element 1: Physical File

***NONE:** There is no associated physical file for this entry.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the physical file with which this entry is associated.

Element 2: Member

***FIRST:** The entry is associated with the first member in the file.

member-name: Specify the name of the physical file member with which this entry is associated.

FORCE

Specifies whether the journal receiver is forced to auxiliary storage after the user entry is written to it.

***NO:** The journal receiver is not forced to auxiliary storage.

***YES:** The journal receiver is forced to auxiliary storage.

Examples

Example 1: Forcing Journal Receivers to Auxiliary Storage

```
SNDJRNE   JRN(JRNLA)  TYPE(AB)
  ENTDTA('PROGRAM COMPLETE')
  FILE(MYLIB/ORDERENT MBR1)  FORCE(*YES)
```

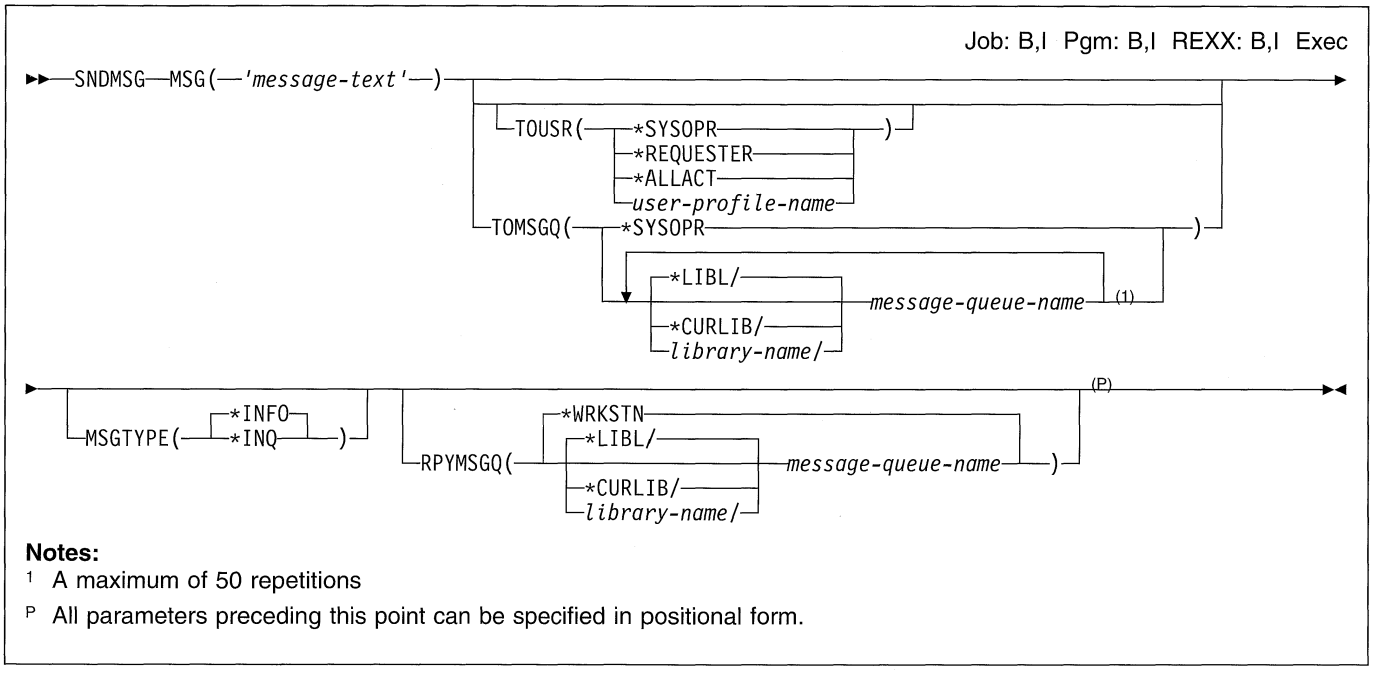
This command places a journal entry of type AB (hex C1C2) with the user-created journal entry data in the current journal receivers attached to journal JRNLA as found by using the library search list. The entry is associated with member MBR1 of file ORDERENT in library MYLIB. The journal receivers are forced to auxiliary storage after the entry has been placed on them.

Example 2: Sending a Journal Entry

```
SNDJRNE   JRN(JRNLA)  TYPE(x'C1F1')
```

This command places a journal entry of type 'A1' (hex C1F1) with no user-created journal entry data in the current journal receivers attached to journal JRNLA as found by using the library search list. The entry is not associated with any physical file member.

SNDMSG (Send Message) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Send Message (SNDMSG) command is used by display station users to send immediate messages from their display stations to one or more message queues. An immediate message is a message that is not predefined and is not stored in a message file. The message can be sent to the system operator, to other display station users, to a user's message queue, to all currently active user's message queues, or to the system history log, QHST. The sender can require a reply from the message receiver. The primary users of this command are display station operators and the system operator.

Restrictions:

1. The user must have object operational and add authorities to the message queue.
2. The user of this command must have *USE authority for the specified message queues and for the libraries in which they are stored.
3. The SNDMSG command only allows a message of up to 512 characters of first-level message text to be sent.
4. This command cannot send inquiry messages (specified by MSGTYPE(INQ)) to multiple message queues.

Required Parameters

MSG

Specifies the message text that is sent. A maximum of 512 characters can be specified. The message must be enclosed in apostrophes if it contains blanks or other special characters.

TOUSR

Specifies that the message is sent to the message queue specified in the user profile for the user named on this parameter. This parameter cannot be used if TOMSGQ is specified.

***SYSOPR:** The message is sent to the system operator's message queue (QSYS/QSYSOPR).

***REQUESTER:** The message is sent to the user profile's message queue for interactive jobs or to the system operator's message queue (QSYS/QSYSOPR) for batch jobs.

***ALLACT:** A copy of the message is sent to the user profile message queue of each user profile with an interactive job currently running. This value cannot be specified if the value *INQ is specified for the MSGTYPE parameter.

user-profile-name: Specify the user profile name of the user to whom the message is sent.

TOMSGQ

Specifies the qualified names of up to 50 message queues to which the message is sent. A message may also be sent to the IBM-supplied log (QHST). This parameter cannot be used if TOUSR is specified.

***SYSOPR:** The message is sent to the system operator's message queue, QSYS/QSYSOPR.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue to which messages are sent.

Optional Parameters

MSGTYPE

Specifies the type of message being sent. Only informational and inquiry message types can be specified.

***INFO:** An informational message is sent.

***INQ:** An inquiry message is sent. The message queue receiving the message can reply to it. Inquiry messages are sent to only one message queue at a time.

RPYMSGQ

Specifies, only if an inquiry message is sent, the qualified name of the message queue to which a reply is sent.

***WRKSTN:** The reply to the message is sent to the display station message queue associated with the sender's display station. For batch jobs, a reply is sent to the system operator's (QSYSOPR) message queue.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue to which the reply is sent.

Examples

Example 1: Sending Message to User Message Queue

```
SNDMSG   MSG('Do you want to update INV now?')
  TOUSR(JONES)  MSGTYPE(*INQ)  RPYMSGQ(SMITH)
```

This command sends a message to the user message queue JONES. When the message is answered, the reply will be sent to the message queue SMITH.

Example 2: Sending Message to System's History Log

```
SNDMSG   MSG('Input errors on PAYROLL cost
  me 1 hour of run time.')  TOMSGQ(QHST)
```

This command is used by the system operator to send an informational message to the system's history log, QHST, through the log's message queue, which has the same name.

Example 3: Sending Message to System Operator

```
SNDMSG   MSG('Please make 2 copies of printer
  file LABORSTAT.')  TOMSGQ(QSYSOPR)
```

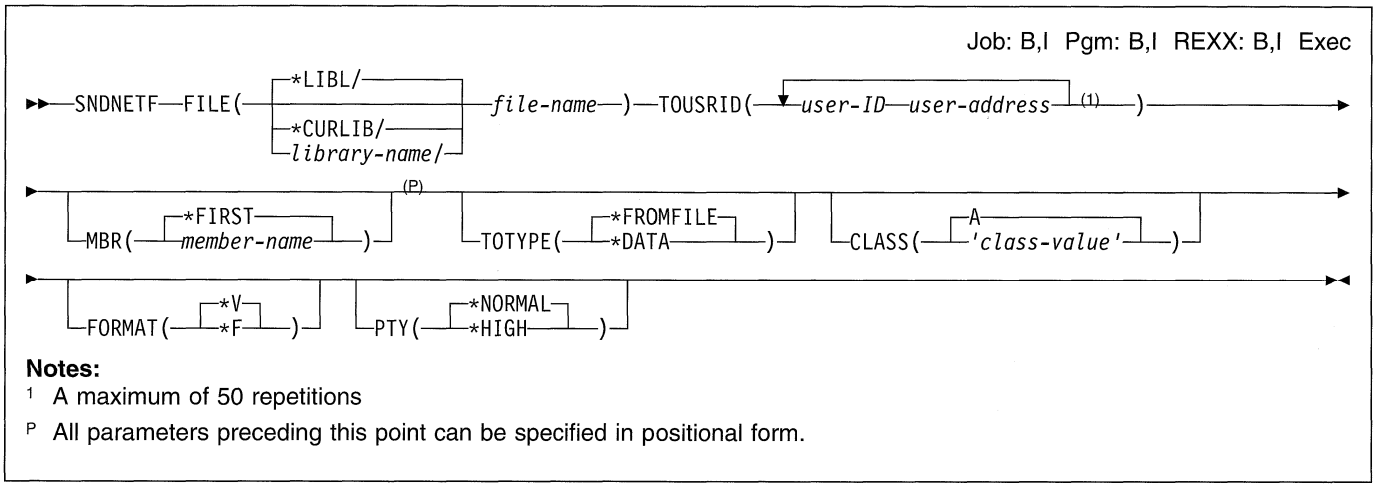
This example shows a typical use of the SNDMSG command by a display station user. The user is sending the message 'Please make 2 copies of printer file LABORSTAT.' to the system operator.

Example 4: Sending Message that Requires a Reply

```
SNDMSG   MSG ('How long will the display stations be
  online today?')  TOMSGQ(*SYSOPR)  MSGTYPE(*INQ)
```

This inquiry message from a display station user requires a reply. The system operator displays the message by using the DSPMSG command and enters the reply on the display. The reply is then sent to the display station user's work station message queue. The display station user enters another DSPMSG command to display the reply.

SNDNETF (Send Network File) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Send Network File (SNDNETF) command sends a save file or a member of a physical database file to another system user through the SNADS network. This command can be used to:

- Send data files to a user.
- Send source files to a user. Source sequence information is kept in the file sent.
- Send other object types stored in a save file to a user.

When the file arrives at its destination, a notification message is sent to both the recipient and sender of the file.

When a source physical file is sent, the source sequence number and change date in positions 1 through 12 of the record are sent with the file. These are kept if the file is received into a source physical file, and are truncated if the file is received into a nonsource physical file. When a file that was originally a nonsource physical file is received into a source physical file, the source sequence numbers are created and placed in front of the records.

Restrictions:

1. The user must be enrolled in the system distribution directory.
2. The maximum size of a file that can be sent using the SNDNETF command is approximately 2 billion bytes.

Required Parameters

FILE

Specifies the name of the file that is sent. The file being sent can be a physical file or a save file; logical files and device files are not allowed. Overrides to the specified file are ignored.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the file that is sent.

TOUSRID

Specifies the two-part user ID of one or more users to whom the file is being sent, or the name of one or more distribution lists containing the user IDs of one or more users to whom the file is being sent. A combination of user IDs and distribution lists can be specified on the same command. Each user ID or distribution list name is specified as a two-part name, and both parts are required.

Note: Depending on the type of work station being used, the internal value for a user identifier may differ from the characters shown by the Display Directory (DSPDIR) command. If the byte-string value specified for the TOUSRID parameter does not match the rules for an internal user identifier value, or if it does not match the internal value for any enrolled user, an error may be reported.

Optional Parameters

MBR

Specifies the member that is sent from the file. A member name is not allowed if the file is a save file.

***FIRST:** The first member in the database file is used.

member-name: Specify the name of the file member that is sent.

TOTYPE

Specifies, when the user sends a source file, whether the sequence numbers and date fields are to be removed from the transmitted copy of the file. The source file is not changed. This parameter is not valid for nonsource files.

***FROMFILE:** The FILETYPE of the source file is used when sending the file. The transmitted file does not change.

***DATA:** The file is sent as a nonsource file. The transmitted copy is sent without sequence numbers and date fields.

CLASS

Specifies the VM/MVS SYSOUT class for distributions bound for a VM/MVS host system.

A: The class value is A.

class-value: Specify a class value. Valid values range from B through Z and 0 through 9.

FORMAT

Specifies the record format in which the network file is transmitted.

***V:** The file is sent using variable-length records with trailing blanks removed from each record.

***F:** The file is sent as fixed-length records with no trailing blanks removed from the records. Specifying this value affects only network files sent to a System/370. This value is not recommended for sending files to another AS/400 system.

Note: Specifying FORMAT(*F) may increase the amount of storage and time required when transmitting the network file.

PTY

Specifies the queuing priority used for this file when it is being routed through a SNADS network.

***NORMAL:** The file is sent with a service level priority of data low, which is used for most data traffic. On an AS/400 system, data low distributions are placed on the normal distribution queue specified for the route.

***HIGH:** The file is sent with a service level priority of data high, which is used for high priority data traffic. On an AS/400 system, data high distributions are placed on the data high distribution queue specified for the route.

Examples

Example 1: Sending a Member

```
SNDNETF  TOUSRID((JONES SYSTEM1))  FILE(EMPLOYEE)
  MBR(PGMR)
```

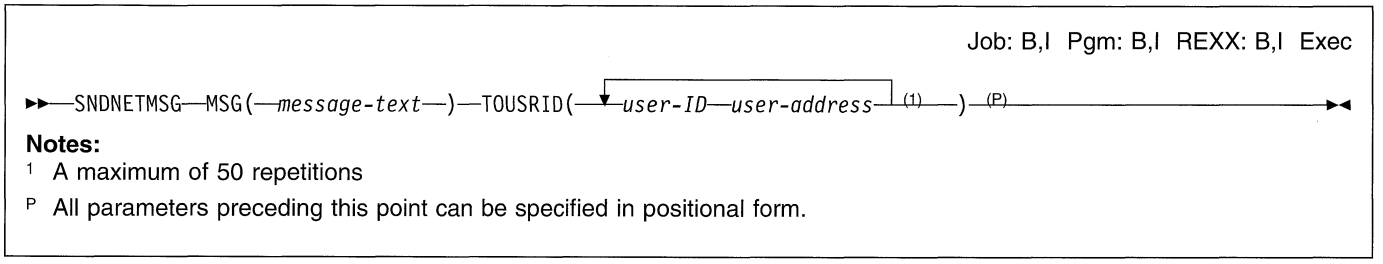
This command sends member PGMR of file EMPLOYEE to the user identified to the network with a user ID of (JONES SYSTEM1). The library list is used to locate the file.

Example 2: Sending a Nonsource File

```
SNDNETF  TOUSRID((JONES SYSTEM2))  FILE(EMPLOYEE)
  MBR(PGMR) TOTYPE(*DATA)
```

This command sends member PGMR of file EMPLOYEE to the user identified to the network with a user ID of (JONES SYSTEM2). The library list is used to locate the file. The file is being sent as a nonsource file removing the sequence numbers and date fields.

SNDNETMSG (Send Network Message) Command



Purpose

The Send Network Message (SNDNETMSG) command sends a message to another system user through the SNADS network. This message is sent as an informational message to the message queue that is defined for the recipient on the receiving system.

Note: The recipient must have a valid message queue specified in his user profile. Messages sent with the SNDNETMSG command are rejected if the recipient does not have a message queue specified in the user profile. The message queue specified in the network attributes is not used. Additional information on specifying a message queue when sending and receiving messages is in the *CL Programmer's Guide*.

Restriction: The user must be enrolled in the system distribution directory. A description of the system distribution directory is in the *Distribution Services Network Guide*.

Required Parameters

MSG

Specifies the message text of the impromptu message that is sent. (An impromptu message is a message that is not stored in a message file.) Specify the message text. The text must be enclosed in apostrophes if it contains blanks or special characters. Up to 256 characters can be specified.

TOUSRID

Specifies one or more user IDs to whom an impromptu message is sent, or the name of one or more distribution lists containing user IDs of users to whom the message is to be sent. A combination of both user IDs and distribution lists can be specified on the same command. Each user ID or distribution list is specified as a two-part name, and both parts are required. Up to 50 user IDs can be specified.

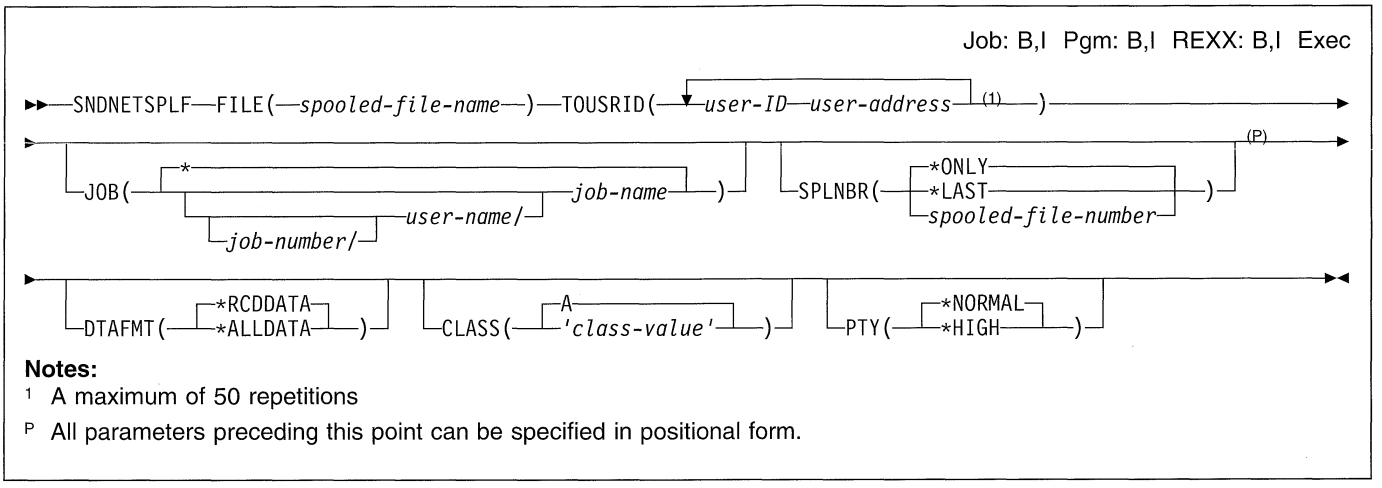
Note: Depending on the type of work station being used, the internal value for a user identifier may differ from the characters shown by the Display Directory (DSPDIR) command. If the byte-string value specified for the TOUSRID parameter does not match the rules for an internal user identifier value, or if it does not match the internal value for any enrolled user, an error may be reported.

Example

```
SNDNETMSG   MSG('Please do not make any more
  changes to the accounts receivable
  files for the next hour.')
  TOUSRID((SMITH SYSTEM2))
```

This command sends the message specified in the MSG parameter to the user identified to the network with a user ID of (SMITH SYSTEM2).

SNDNETSPLF (Send Network Spooled File) Command



Purpose

The Send Network Spooled File (SNDNETSPLF) command sends a spooled file to another system user on the Systems Network Architecture Distribution Services (SNADS) network. The file is placed on an output queue that is specified in the user profile of the user to whom the spooled file was sent.

When the file arrives at the destination system, a message is sent to both the recipient and sending user notifying them of the arrival of the spooled file.

Restrictions:

1. The user must be enrolled in the system distribution directory to run this command. The sender must have read, add, and delete authority to the receiving output queue when sending to user on the same system.
2. One of the following must be true:
   - The requester is the creator of the file.
   - The requester has *READ authority to the output queue on which the file resides, and DSPDTA(*YES) was specified on the CRTOUTQ command.
   - The requester has *SPLCTL special authority.
   - The requester has *JOBCTL special authority, and the output queue on which the file resides has OPRCTL(*YES) specified on the CRTOUTQ command.
   - The output queue has DSPDTA(*YES) specified on the CRTOUTQ command.
   - The requester has owner authority to the output queue on which the file resides and the queue had AUTCHK(*OWNER) and DSPDTA(*YES) or DSPDTA(*NO) specified on the CRTOUTQ command.
   - The requester has *READ, *ADD, and *DELETE authority to the output queue on which the file resides and the queue has AUTCHK(*DTAAUT) and DSPDTA(*YES) or DSPDTA(*NO) specified on the CRTOUTQ command.
3. DTAFMT(*RCDDATA) must be used when sending a spooled file to a release prior to Version 1 Release 3 Modification 0 (V1R3).

Required Parameters

FILE

Specifies the name of the spooled file that is sent to the specified user. The file name is the name of the device file that was used by the program to produce the spooled file. Specify the name of the spooled file.

TOUSRID

Specifies the user ID of one or more users to whom the spooled file is sent, or the name of one or more distribution lists containing the user IDs of one or more users to whom the spooled file is sent. A combination of user IDs and distribution lists may be specified on the same command. Each user ID or distribution list name is specified as a two-part user name, and both parts are required. The users in the distribution list may be either remote or local.

Note: Depending on the type of work station being used, the internal value for a user identifier may differ from the characters shown by the DSPDIR command. If the byte-string value specified for the TOUSRID parameter does not match the rules for an internal user identifier value, or if it does not match the internal value for any enrolled user, an error may be reported.

Optional Parameters

JOB

Specifies the name of the job that created the spooled file whose data records are sent. If no job qualifier is given, all jobs currently in the system are searched for the simple job name.

A job identifier is a special value or a qualified name with up to three elements. For example:

```
*
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** The job that issued this command is the job that created the spooled file.

job-name: Specify the name of the job that created the spooled file.

user-name: Specify the name of the user of the job that created the spooled file.

job-number: Specify the number of the job that created the spooled file.

SPLNBR

Specifies the number of the spooled file from the job whose data records are to be copied. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ONLY:** One spooled file from the job has the specified file name. The number of the spooled file is not necessary. If *ONLY is specified and more than one spooled file has the specified file name, a message is sent.

***LAST:** The spooled file with the highest number and the specified file name is used.

spooled-file-number: Specify the number of the spooled file having the specified file name whose data records are copied.

DTAFMT

Specifies the format in which to send a spooled printer file. This parameter is valid only for spooled printer files; it is not valid for diskette files.

If DTAFMT(*RCDDATA) is specified, then a spooled file that contains special device requirements cannot be sent.

Note: The device requirements are listed as part of the attributes for the spooled file. To view the device requirements, use the WRKSPLFA command or the attributes option on the WRKSPLF command. If any of the device requirements are attributes of the file (if any of the device requirements on the display have a 'Y'), the spooled file cannot be sent. Either specify DTAFMT(*ALLDATA) or copy the spooled file to a database file using the Copy Spooled File (CPYSPLF) command and then use the Send Network File (SNDNETF) command to send the file as a data file.

If DTAFMT(*RCDDATA) is specified, the following attributes of the spooled file are kept:

- File name
- Number of copies
- Characters per inch
- Drawer
- Form type
- Lines per inch
- Print text
- Page length
- Page rotation
- Page width
- Font name
- Diskette label
- Diskette creation and end dates
- Diskette code type
- Diskette exchange type

***RCDDATA:** The spooled printer file is converted to a format acceptable for releases prior to V1R3 of the AS/400 system, including System/36, System/38, and System/370. A limited set of attributes are sent.

***ALLDATA:** The spooled printer file is sent in the same format as it is stored on the spool. This value is only compatible with V1R3 or later releases of the AS/400 system.

Notes:

1. If a file will be printed on the receiving system, it must be printed on the same type of printer as it was intended to be printed on the source system.
2. Spool files that require advanced function printing (AFP) resources may print differently on the receiving system. To assure that spool files are printed in the same manner, the following must occur:
   - Before using this command to send a file to a user on the same system, make sure that the libraries containing non-IBM supplied AFP resources are in the library list.
   - Before using this command to send a file to a different user or system, make sure that the libraries containing non-IBM supplied AFP resources are in the initial library list of the user receiving the spooled files.
3. Use this format to send *LINE, *AFPDS, and *AFPDSLINE printer type device files to System 370.

CLASS

Specified the VM/MVS SYSOUT class for distributions bound for a VM/MVS host system.

A: The class is A.

class-value: Specify a distribution class value. Valid values range from B through Z and 0 through 9.

PTY

Specifies the queuing priority used for this spooled file when it is being routed through a SNADS network.

***NORMAL:** The spooled file is sent with a service level priority of data low, which is used for most data traffic.

On an AS/400 system, data low distributions are placed on the normal distribution queue specified for the route.

***HIGH:** The spooled file is sent with a service level priority of data high, which is used for high priority data traffic. On an AS/400 system, data high distributions are placed on the data high distribution queue specified for the route.

Examples

Example 1: Sending a Spooled File

```
SNDNETSPLF  FILE(QPRINT)  TOUSRID((JDE SYS1))
  JOB(142857/PAPER/PRINT)  SPLNBR(*LAST)
  DTAFMT(*ALLDATA)
```

This command sends the last (most recently created) copy of spooled file QPRINT from job 142857/PAPER/PRINT to the user with a user ID of JDE SYS1. All spooled file functions will be sent.

Example 2: Sending Print Attributes

```
SNDNETSPLF  DTAFMT(*RCDDATA)
```

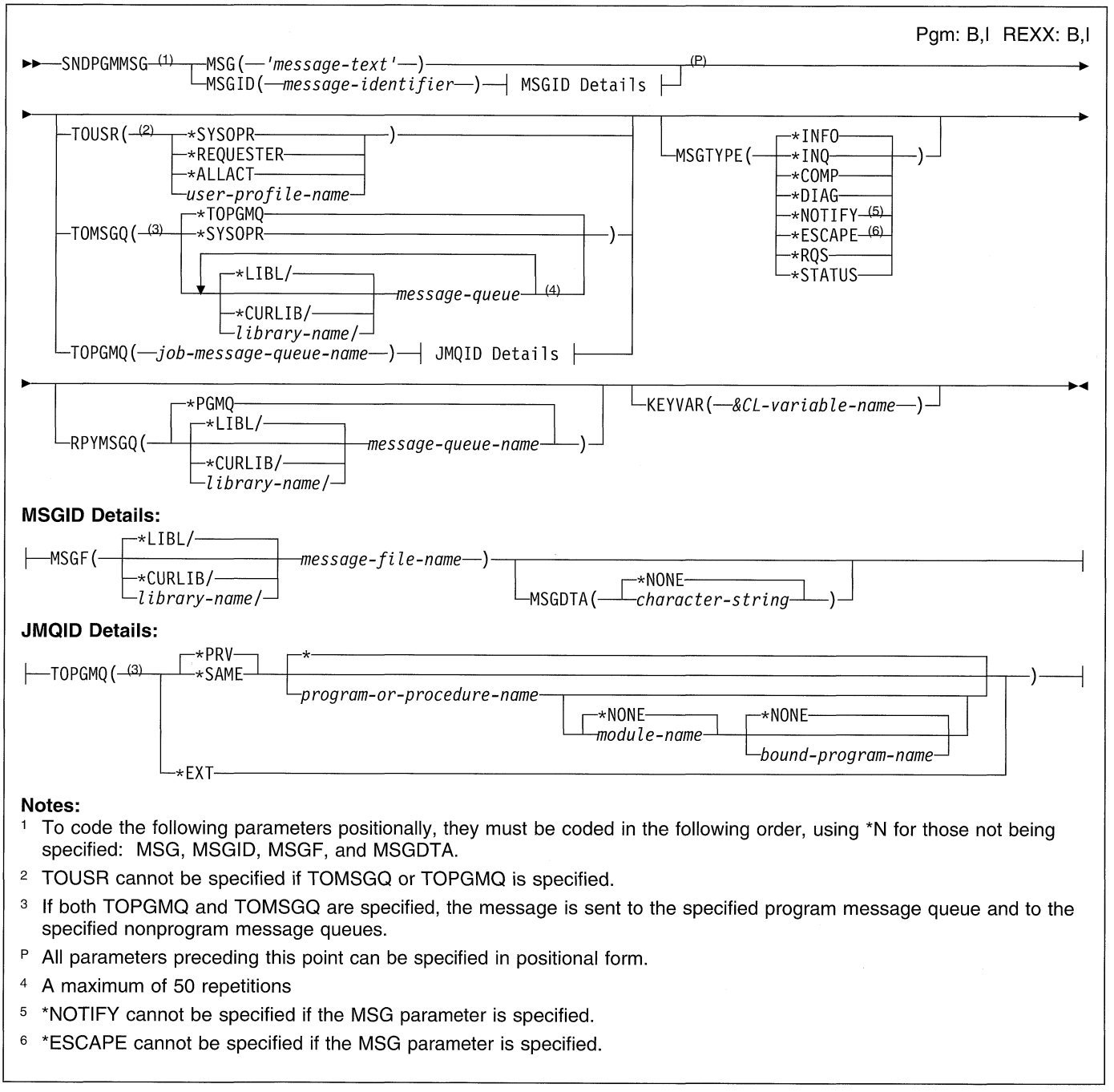
This command sends a limited set of print attributes.

Example 3: Sending All Print Attributes

```
SNDNETSPLF  DTAFMT(*ALLDATA)
```

This command sends all print attributes. *ALLDATA is only valid when it is sent from one AS/400 to another AS/400, and both systems are Release 3.0 or higher.

SNDPGMMSG (Send Program Message) Command



Notes:

1 To code the following parameters positionally, they must be coded in the following order, using *N for those not being specified: MSG, MSGID, MSGF, and MSGDTA.

2 TOUSR cannot be specified if TOMSGQ or TOPGMQ is specified.

3 If both TOPGMQ and TOMSGQ are specified, the message is sent to the specified program message queue and to the specified nonprogram message queues.

P All parameters preceding this point can be specified in positional form.

4 A maximum of 50 repetitions

5 *NOTIFY cannot be specified if the MSG parameter is specified.

6 *ESCAPE cannot be specified if the MSG parameter is specified.

Purpose

The Send Program Message (SNDPGMMSG) command sends a message to a named message queue or to a call message queue. A call message queue can be the *EXT external message queue or a message queue associated with a call stack entry. Each time a program or procedure is called a new message queue is associated with its call stack entry. The message queue is identified by the name of its associated program or procedure.

A program can send a message to its own message queue or to a message queue that is associated with a different call stack entry.

This command can send both exception and non-exception messages. When an exception message is sent, processing of the sending program is interrupted until action is taken on the exception. Depending on the action taken, processing of the sending program may resume. If allowed to resume, processing continues in the sending program at the CL command immediately following the SNDPGMMSG command. When a non-exception message is sent, pro-

cessing of the sending program is not interrupted. After the message is sent, the sending program continues to run at the CL command immediately following the SNDPGMMSG command.

When a program or procedure monitors for an exception message and an exception message is sent to the call stack entry in which that program or procedure is running, the related exception handler is called and run by the system. Therefore, a program or procedure can identify when an exception message is sent to the call stack entry in which it is running. There is no automatic notification for non-exception messages.

The message can be an immediate or a predefined message that is stored in a message file. Substitution data can be specified in the MSGDTA parameter (as a single character string containing one or more concatenated message data fields) to replace the substitution variables in the message. The message can be sent to the job's *EXT message queue, to a message queue associated with a call stack entry, to one or more nonprogram message queues (such as a user profile message queue), to the system operator message queue, or to the system history log, QHST.

Restrictions:

1. This command is valid only in CL programs.
2. The SNDPGMMSG command allows a message of up to 512 characters to be sent. However, if the message is sent to the *EXT message queue of an interactive job, only 76 characters are shown on the Display Program Messages display. If the message is sent to a user's, work station's, or the system operator's message queue, the Display Message (DSPMSG) command allows all 512 characters to be displayed.

Required Parameters

MSG

Specifies the message text that is sent. A maximum of 512 characters can be specified. The string must be enclosed in apostrophes if special characters (including blanks) are used. If a value is specified on this parameter, the MSGID, MSGF, and MSGDTA parameters cannot be specified.

MSGID

Specifies the message identifier of a message description whose predefined message is being sent by the program to a message queue. If MSGID is specified, the MSG parameter cannot be specified.

MSGF

Specifies the qualified name of the message file that contains the predefined message being sent. This parameter is required if MSGID is specified.

The name of the message file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-file-name: Specify the name of the message file that contains the predefined message.

Optional Parameters

MSGDTA

Specifies the character string, or a CL variable that contains a character string, containing one or more substitution values that are used as message data fields within the predefined message. The substitution values take the place of the substitution variables that were defined in the message text when the message was defined.

Note: MSGDTA is not valid if MSG is specified.

Rules for Replacing CL Variables with Character Strings

- Multiple values must be concatenated to form a single character string (more information is in the *CL Programmer's Guide*).
- If more than one substitution variable exists in the message, the specified character string must be the same length as the sum of the message data fields defined in the message.
- The length of the entire character string of concatenated substitution values cannot be greater than 512 characters. More information about the length of individual message data fields is in the FMT parameter description in the ADDMSGD (Add Message Description) command.
- Each substitution value, specified in the same order in the string as defined in the FMT parameter of the ADDMSGD command, must be specified but only if its associated variable was defined.
- If the length of the message data that is passed to a substitution variable is shorter than the length specified for the FMT parameter of the ADDMSGD command, the substitution value becomes a null field.
- If the length of the message data passed to a substitution variable is longer than the length specified for ADDMSGD FMT, the message data is truncated.

  Note: If the last character in the string passed to the substitution variable is blank (for the format types *BIN, *ITV, or *DTS, a blank is indicated by the hex value 40), the system truncates the blank. If the resulting string is shorter than the length specified on the FMT

parameter of the ADDMSGD command, the substitution variable becomes a null field.

To prevent the system from truncating the blanks, concatenate a nonblank character (for the format type *BIN, *ITV, and *DTS, add value other than 40) to the string after the blanks. The system retains the blanks and truncates the additional character.

***NONE:** No message data is supplied for the message.

character-string: Specify either the character string that gives the substitution values in the specified predefined message sent by the program, or the name of the CL variable that contains the character string.

If the message contains more than one substitution variable, the character string containing the replacement values must be coded with the proper fill characters (zeros for numeric values, and blanks for character values) to position the values properly in the string to match the concatenated fields used to create the single string. If the string contains special characters (including blanks), it must be enclosed in apostrophes. The null value *N cannot be specified for a substitution value.

For example, if a predefined message contains the following three substitution variables:

Variable	Format Type	Length	Expected Values
&1	*CHAR	3 characters	Yes or no
&2	*CHAR	4 digits	4 digits
&3	*CHAR	8 characters	One or two words

the following valid character strings can be specified:

```
MSGDTA('Yes0007not done')
MSGDTA('No 0285finished')
MSGDTA(&DTA)
```

Note that the variable &2 is defined as a character variable, not as a decimal variable. If a character string is to be used instead of a CL variable to specify the substitution values in the MSGDTA parameter, the associated substitution variables must be defined (in the FMT parameter of the ADDMSGD command) as character variables to prevent invalid results in the text of the message that is sent.

The last example shows, instead of the actual character string, the CL variable &DTA, whose value is used as the message data. &DTA might have been declared in the sending program as:

```
DCL   VAR(&DTA)  TYPE(*CHAR)
  LEN(15)  VALUE('YES0175not done')
```

The character string, specified as the initial value for &DTA, contains the three concatenated substitution values `YES`, `0175`, and `not done`, which are to be sent as message data.

TOUSR

Specifies that the message is to be sent to the message queue specified in the user profile for the user named on this parameter. This parameter cannot be used if TOMSGQ or TOPGMQ is specified.

***SYSOPR:** The message is sent to the system operator's message queue, QSYS/QSYSOPR.

***REQUESTER:** The message is sent to the user profile's message queue for interactive jobs or to the system operator's message queue (QSYS/QSYSOPR) for batch jobs.

***ALLACT:** A copy of the message is sent to the user profile message queue of each user profile with an interactive job currently running. This value cannot be specified if an inquiry message is being sent (the value *INQ is specified in the MSGTYPE parameter).

user-profile-name: Specify the name of the user profile of the user to whom the message is sent.

TOMSGQ

Specifies the qualified names of one or more nonprogram message queues to which a message is sent by the program. This parameter cannot be used if TOUSR is specified.

***TOPGMQ:** The message is sent only to the call message queue specified on the TOPGMQ parameter.

***SYSOPR:** The message is sent to the system operator's message queue, QSYSOPR.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the names of the message queues to which the message is sent. Up to 50 names can be specified.

TOPGMQ

Specifies the call message queue to which the message is sent. The call message queue can be the *EXT external queue or a program or qualified procedure message queue.

Element 1: Relationship

Element 1 of this parameter specifies whether the message queue is associated with a program or procedure identified by Element 2, or if it is associated with the caller of the program or procedure.

***PRV:** The message is sent to the message queue of the program or procedure that called the program or procedure identified by Element 2.

***SAME:** The message is sent to the message queue of the program or procedure identified by Element 2.

Element 2: Program or Qualified Procedure

Element 2 of this parameter has three items. Item 1 specifies the program or procedure of the job message queue. Items 2 and 3 specify the module name and the bound program name, which can be used to qualify the procedure name.

Item 1: Program or Procedure Name

***:** Identifies the program running the SNDPGMMSG command.

program-or-procedure-name: Specify the name of the program or procedure used.

If Item 1 identifies a program, the name specified can be a maximum of 10 characters. If Item 1 identifies a procedure, the name specified can be a maximum of 256 characters. The system begins its search for the specified program or procedure name with the most recently called program or procedure.

The procedure name alone may not identify the correct procedure. Several different procedures with the same name can run in a job. To further identify a procedure, the name specified can be qualified by a module name, or by both a module name and a bound program name.

Item 2: Module Name

The module name qualifier identifies the module into which the procedure was compiled.

***NONE:** No module name is specified.

module-name: Specify the module name to be used as a qualifier for the specified procedure name (modules are associated only with procedures). The module name can be a maximum of 10 characters.

If a module name is not specified but a bound program name is, *NONE must be specified in the module name position.

Item 3: Bound Program Name

The bound program name qualifier identifies the program to which the procedure was bound.

***NONE:** No bound program name is specified.

bound-program-name: Specify the bound program name to be used as a qualifier for the specified procedure name (and module name if specified). The bound program name can be a maximum of 10 characters.

Other Single Values

***EXT:** The message is sent to the external message queue of the job. The external message queue is used to communicate with the external requester of the job, such as a display station user. *INQ messages that are sent to *EXT wait for 24 hours before the default reply is sent.

Messages sent to this queue can be 512 characters in length, but only 76 characters of text are shown on the Program Messages display.

MSGTYPE

Specifies the message type assigned to a message sent by this program.

Notes:

1. Inquiry messages can be sent only to the external queue or to a named message queue specified on the TOUSR or TOMSGQ parameters.
2. Completion, diagnostic, escape, notify, and status messages can be sent only to a call message queue.
3. Escape messages cannot be sent to the external message queue.

***INFO:** The message is sent as an informational message.

***INQ:** The message is sent as an inquiry message.

***COMP:** A completion message is sent to a call message queue. A completion message indicates the status of the work that is successfully performed.

***DIAG:** A diagnostic message is sent to a call message queue. Diagnostic messages provide information about errors detected by this program. The errors are either in the input sent to it, or are those that occurred while it was running the requested function. An escape or notify message is also sent to inform the receiving program or procedure of the diagnostic messages that are on its message queue.

***NOTIFY:** A notify exception message is sent to a call message queue. A notify message describes a condition for which corrective action must be taken before the sending program can continue. A reply message is sent back to the sending program. After corrective action is taken, the sending program can resume running and can receive the reply message from its message queue.

***ESCAPE:** An escape exception message is sent to a call message queue. An escape message describes an irrecoverable error condition. The sending program does not continue to run.

***RQS:** A request message is sent to a call message queue. A request message allows request data received from device files to pass from this program to another program or procedure. An immediate message, specified by the MSG parameter, must be used to send the request.

***STATUS:** A status exception message is sent to a call message queue. The status message describes the status of work performed by the sending program. The first 28 characters of message data in the MSGDTA parameter are used as the comparison data for message monitors (established by the Monitor Message (MONMSG) command). If the status exception message is not being monitored, control is returned to the sending

program. If a status message is sent to the external message queue of an interactive job, the message is shown on line 24, processing continues, and no response is required.

Note: This value cannot be specified if the MSQ parameter is specified.

RPYMSGQ

Specifies, for inquiry and notify messages only, the name of the call message queue or the qualified name of the nonprogram message queue to which a reply message is sent.

***PGMQ:** The reply to an inquiry or notify message is sent to the message queue associated with the call stack entry of the program or procedure using this command.

The name of the nonprogram message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the nonprogram message queue to which the reply is sent.

KEYVAR

Specifies the name of the CL character variable, if any, that contains the message reference key that identifies the message sent by the program containing this SNDPGMMSG command. The message reference key is assigned by the system when the message is sent and is placed in the variable specified by KEYVAR. The message key can then, for example, be used in the MSGKEY parameter of the RCVMSG command to receive the reply to an inquiry message being sent by this command. The message reference key can be used by the program specified on the TOPGMQ parameter for building message subfiles. The CL variable is the name of the field for which the SFLMSGKEY keyword is specified in the DDS for the message subfile.

If a message is being sent to a message queue associated with a call stack entry, KEYVAR refers to that message queue (specified by the TOPGMQ parameter). If *INQ or *NOTIFY is specified for the MSGTYPE parameter, KEYVAR refers to the message queue specified in the RPYMSGQ parameter. In all other cases, KEYVAR refers to the message queue specified in the TOMSGQ parameter.

If an inquiry message has been sent to the *EXT message queue with a value specified for the KEYVAR parameter, the message reference key returned is that of the sender's copy message, which is located on the reply message queue. The reply message queue defaults to the program message queue of the program that sent the inquiry message.

Any type of message can be assigned a key when it is being sent to a program message queue. For messages sent to a nonprogram message queue, message reference keys are available for inquiry (*INQ) messages only. If another message type is sent to a nonprogram queue, no message key is available and blanks are returned for KEYVAR.

The variable must be a character variable having a length of 4 characters. If KEYVAR is not specified and a reply is required, it can be received by the program in FIFO order.

Examples

Example 1: Specifying Substitution Values

```
SNDPGMMSG   MSGID(UIN0023)  MSGF(INV)
  MSGDTA('50 100')  TOPGMQ(*EXT)
```

This command sends the message identified as UIN0023, which is stored in message file INV, to the external message queue of the job (the Display Program Messages presents the message at a display station). The data, which contains two substitution values specified in the MSGDTA parameter, is sent with the message. This data can then be used as substitution values when the message is received, or it can be used as data to be dumped, depending on how the message UIN0023 is defined in the message file. Assuming that the variables &1 and &2 have been defined in the message file as character variables, each 3 characters long, and that the first-level message text of the message UIN0023 is: 'Requested item decreased by &1; current balance &2.' The message text sent is: 'Requested item decreased by 50; current balance 100.'

Example 2: Sending an Inquiry Message

```
SNDPGMMSG   MSG('Mount payroll checks in printer
  before continuing')  MSGTYPE(*INQ)
  TOMSGQ(*SYSOPR)
```

This command sends an inquiry message to the system operator. The operator looks at the message that was sent by using the DSPMSG command and responds to the message directly on that display. A Receive Message (RCVMSG) command is used in the program to accept the operator's response.

Example 3: Sending an Escape Message

```
SNDPGMMSG   MSGID(USR0001)  MSGF(USRMSGR)
  TOPGMQ(*PRV  *)  MSGTYPE(*ESCAPE)
```

This command is an example of how a message could be sent to the caller of a program or procedure to cause an abnormal end. The message USR0001 could indicate that an invalid code was passed (such as a nonnumeric code when numeric is required). Because the message being sent is an escape message, the program or procedure that is sending the message cannot be resumed. The values *PRV

and * did not have to be coded on this command because they are the default values on the TOPGMQ parameter.

Example 4: Sending an Escape Message to an ILE Procedure

```
SNDPGMMSG   MSGID(USR0001)  MSGF(USRMSGR)
  TOPGMQ(*SAME ACCOUNT_FINAL_TOTALS)  MSGTYPE(*ESCAPE)
```

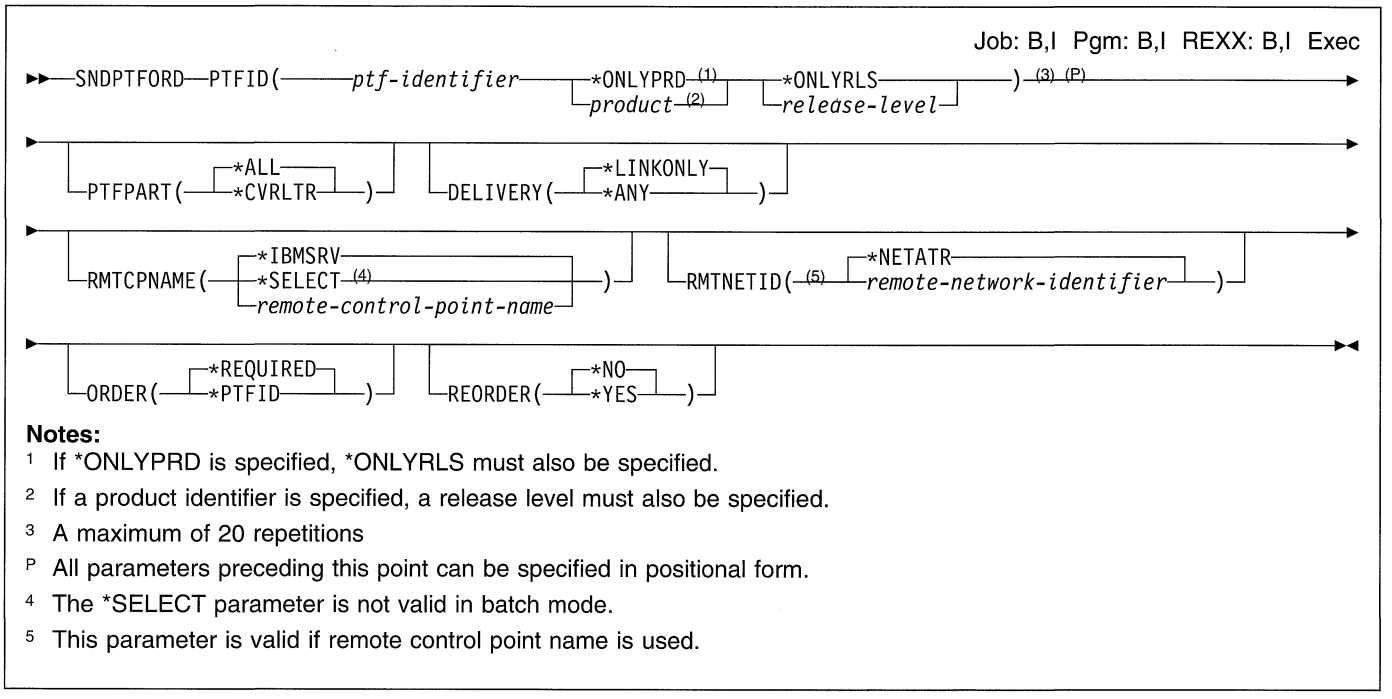
This command sends a message to an ILE procedure. In this example, the call stack entry identifier is more than 10 characters. Since no qualifier is specified, the actual module name and bound program name associated with the procedure are not used in finding the procedure. The escape exception message is sent to the message queue associated with ACCOUNT_FINAL_TOTALS because *SAME is specified for Element 1.

Example 5: Sending an Escape Message using Qualifiers

```
SNDPGMMSG   MSGID(USR0001)  MSGF(USRMSGR)
  TOPGMQ(*PRV FIRST_QTR_SUMMARY SUMQTRS REPORTS)
  MSGTYPE(*ESCAPE)
```

This command sends an escape exception message to the caller of the procedure FIRST_QTR_SYMMARY. The procedure is qualified by the module name SUMQTRS and the bound program name REPORTS. The escape exception message interrupts the sending program and the sending program is not resumed.

SNDPTFORD (Send Program Temporary Fix Order) Command



Notes:

1 If *ONLYPRD is specified, *ONLYRLS must also be specified.

2 If a product identifier is specified, a release level must also be specified.

3 A maximum of 20 repetitions

P All parameters preceding this point can be specified in positional form.

4 The *SELECT parameter is not valid in batch mode.

5 This parameter is valid if remote control point name is used.

Purpose

The Send Program Temporary Fix Order (SNDPTFORD) command allows the user to send an order for a specific fix, a list of fixes, or a fix package to a service provider.

The user can order the following:

- A corrective or preventive PTF package
- Summary information for available PTFs
- Preventive Service Planning (PSP) information

Restriction: This command is shipped with public *EXCLUDE authority and the QSRV and QSRVBAS user profiles have private authority to use the command.

Required Parameter

PTFID

Specifies the list of PTFs being ordered.

Element 1: PTF Identifier

ptf-identifier: Specify the PTF identifier.

Some PTFs must be ordered individually or within a list of PTFs with the same prefix and not as part of a general list. Two PTF numbers are reserved and cannot be ordered.

The following chart shows special classes of PTFs and ordering restrictions.

PTF Prefix	Individual Order	Group Order	Cover Letter	Reserved PTF ID
IInnnnn		X	X	
MF98nnn		X	X	
MF99nnn				X
SF95nnn				X
SF97vrm	X		X	
SF98nnn		X	X	
SF99vrm	X			
SHnnnnn	X			

The following PTFs fall into the special classes:

IInnnnn Information APAR

MF98nnn Hardware preventive service planning information

MF99nnn Reserved PTF

SF95nnn Reserved PTF

SF97vrm PTF summary

SF98nnn Software preventive service planning information

SF99vrm Fix package

SHnnnnn Manufacturing, accounting and production information control system (MAPICS*) and construction management and accounting system (CMAS) PTFs

```
n is the PTF sequence number
v is the version for the package requested
r is the release for the package requested
m is the modification level
  for the package requested
```

Element 2: Product

***ONLYPRD:** The PTF order is for products that are installed or supported on your system.

product: Specify the product. The PTF order is limited to the product entered.

Element 3: Release Level

***ONLYRLS:** The PTF order is for the release levels of the products that are installed or supported on your system.

release-level: Specify the release level in the format VxRxMx, where Vx is the version number, Rx is the release number, and Mx is the modification number.

Optional Parameters

PTFPART
Specifies whether PTFs or cover letters are being ordered.

***ALL:** PTFs and cover letters are being ordered.

***CVRLTR:** Cover letters only are being ordered.

DELIVERY
Specifies how the PTFs are delivered.

***LINKONLY:** PTFs are delivered by the electronic customer support service link only.

***ANY:** PTFs are delivered by an any available method. The service link is used for most PTFs. PTFs that are too large for the service link are sent on a tape.

RMTCPNAME
Specifies the destination of the service provider to whom the service request is sent.

***IBMSRV:** The service request is sent to IBM service support.

***SELECT:** A list of service providers is shown so the user can select the destination for the service request.

remote-control-point-name: Specify the name of the control point that is the destination of the request.

RMTNETID
Specifies the remote name of the service provider's network.

***NETATR:** The service provider is in the local network.

remote-network-identifier: Specify the network name of the service provider to whom the request is being sent.

ORDER
Specifies the level of fixes that are being requested.

***REQUIRED:** The PTF ordered and its requisites are being requested.

***PTFID:** The specific PTF ordered is the one being requested. No requisites are sent.

REORDER
Specifies whether a PTF that is currently installed or ordered is ordered again. If ORDER(*REQUIRED) is specified, this parameter also applies to requisites.

***NO:** PTFs that are already on order or installed are not reordered.

***YES:** PTFs that are already installed or on order are reordered.

Note: A PTF is not reordered if a save file is available on the system.

Examples

Example 1: Sending a Request

```
SNDPTFORD  PTFID(1234567 1234600)
```

This command sends a request for PTF numbers 1234567 and 1234600.

Example 2: Sending a Request

```
SNDPTFORD  PTFID(1234567)  DELIVERY(*ANY)  ORDER(*PTFID)
```

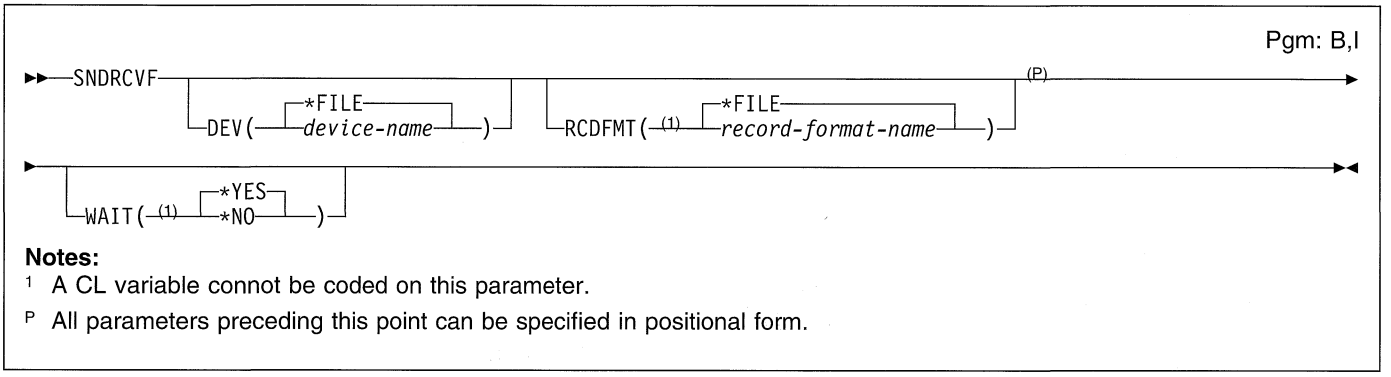
This command sends a request for PTF 1234567. The PTF does not have to be sent over the service link, and no requisites are sent with the PTFs.

Example 3: Sending a Request

```
SNDPTFORD  PTFID(SF99130)
```

This command requests that the latest PTF cumulative package be sent for Release 3.0.

SNDRCVF (Send/Receive File) Command



Notes:
1 A CL variable connot be coded on this parameter.
P All parameters preceding this point can be specified in positional form.

Purpose

The Send/Receive File (SNDRCVF) command is used by a control language (CL) program to send data to and receive data from a device that is being used interactively by a user. The data is passed between the program in which the SNDRCVF command is used and the display device identified in the command. The data is passed using the display device file that was declared in the program. (A Declare File (DCLF) command included in the source used to compile the program was used to declare the file.) The data for each send/receive operation is passed as one record in a format identified by the RCDFMT parameter of this command (the format is defined in the data description specifications (DDS)). One CL variable is used for each field of the record format to pass the data. The CL variables used (including DDS indicators) are declared implicitly.

Of the record formats specified in the DCLF command, only one can be specified in each SNDRCVF command. If the device file has not been opened, it is opened by this command. The file and record format specified in this command can be overridden by an Override with Display File (OVRDSPF) command if that command is entered before the file is opened. However, care should be taken that the fields in the overriding record format correspond to the CL variables declared in the program.

Restriction: This command is valid only for display files within a CL program. It cannot be used with database files.

Optional Parameters

DEV

Specifies the name of the display device to which the CL program's data is sent and from which the user's data is received. A CL variable can be specified on this parameter so that the device name can be changed without changing the command.

***FILE:** The data is sent to and received from the device associated with the device file (the device file that was declared in the FILE parameter of the DCLF command). If more than one device name is specified in the device file, *FILE cannot be specified.

device-name: Specify the name of the device or the name of the CL variable that contains the name of the device to which the CL program sends data and from which it receives data. If a CL variable name is used in this parameter, only one SNDRCVF command is needed in the program to receive data from several devices.

RCDFMT

Specifies the name of the record format that is used to pass the data between the CL program and the user. The format contains all the fields in the record. This parameter must be coded with a record format name if there is more than one record format in the device file; *FILE cannot be coded if there is more than one. SNDRCVF ignores the INVITE DDS keyword.

***FILE:** Only one record format is in the device file, which is used to send the data to and receive the data from the user.

record-format-name: Specify the name of the record format in which the data is sent to and received from the user. A CL variable name cannot be used here to specify the record format name.

WAIT

Specifies whether the CL program waits to receive the data from the user's device or continues to process the commands that follow this SNDRCVF command. If WAIT(*NO) is specified, the program must issue a WAIT command later in the program to complete the input operation.

Note: A CL variable cannot be coded on this parameter.

***YES:** The program waits until the input operation from the device is completed before the next command is processed.

***NO:** The program does not wait for the input data; it continues to process commands until a WAIT command is reached later in the program.

Examples

Example 1

```
DCLF   FILE(MENU1)
  •
  •
  •
SNDRCVF
```

This command sends and receives user data by way of the device file MENU1. Only one record format exists in the file. The device used is specified in the file.

Example 2

```
DCLF   FILE(SCR)  RCDFMT(REC8)
  •
  •
  •
SNDRCVF   RCDFMT(REC8)
```

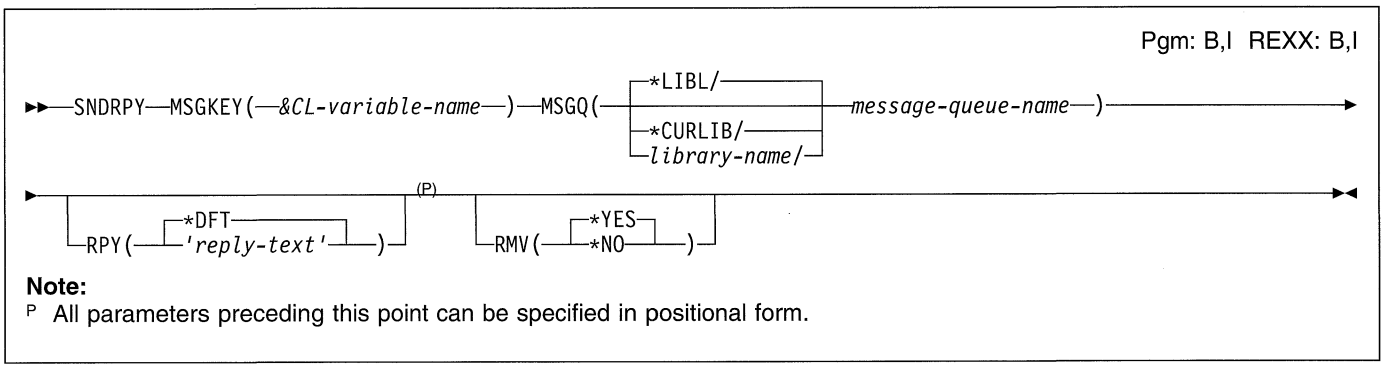
The CL program sends data to a user and receives data for the user who is using the device named in the device file (*FILE is assumed because DEV is not specified). The data is passed in the format specified by REC8 record format in the device file named SCR. The CL program waits for the user data before continuing.

Example 3

```
DCLF   FILE(DF1)  RCDFMT(REC8)
  •
  •
  •
SNDRCVF   DEV(&DN)  RCDFMT(REC8)  WAIT(*NO)
  •
  •
  •
WAIT   DEV(&DN)
```

This command sends and receives user data by way of the device file named DF1. Using the record format REC8, the CL program passes data between itself and the user who is at the device named in the variable &DN, but it does not wait for a response to come back. If the program sends and receives data from several devices, the same SNDRCVF command can be used. Only the device specified by &DN for the DEV parameter must be changed. A WAIT command for each device must be issued later in the program to ensure that all the devices respond.

SNDRPY (Send Reply) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Send Reply (SNDRPY) command sends a reply message to the sender of an inquiry message. The message that is answered is the one having the specified message reference key that was received at the specified message queue. If the specified message queue is not allocated to the job in which this command is entered or to any other job, it is implicitly allocated by this command for the duration of the command.

Required Parameters

MSGKEY

Specifies the name of the control language (CL) variable in the program that contains the message reference key of the message that the reply answers.

MSGQ

Specifies the qualified name of the message queue that received the inquiry message being answered.

| The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue that received the inquiry message.

Optional Parameters

RPY

Specifies the reply that the program sends as a response to the inquiry message.

***DFT:** The default reply to the inquiry message is sent. If no default reply is specified in the message description of the inquiry message, the system default reply, *N, is used.

'reply-text': Specify the text (enclosed in apostrophes if it contains blanks or special characters) or a CL variable that contains the text that is sent as the program's reply to the inquiry message. The number of characters and their format are defined by the validity specifications given in the Add Message Description (ADDMSGD) command for the specified inquiry message. However, if no validity specifications are specified for replies in the ADDMSGD command, as many as 132 characters can be used in the reply text.

RMV

Specifies whether the inquiry message and its reply are removed from the specified message queue.

***YES:** The message and its reply are removed from the message queue when the reply is sent.

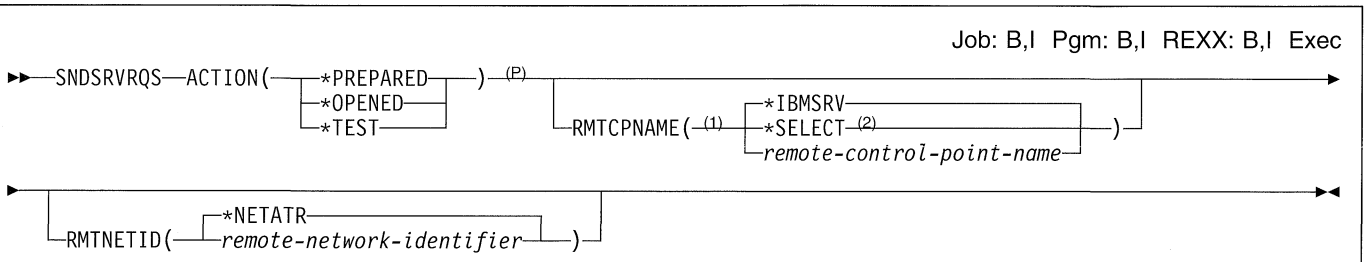
***NO:** The message and its reply are retained in the message queue. The inquiry message cannot be replied to more than once, but it can be received or displayed several times.

Example

```
SNDRPY   MSGKEY(&KEY)  MSGQ(SMITH)  RPY(YES)
```

This command sends a reply of YES to the message whose reference key is specified by &KEY, which was received at message queue SMITH. Because the reply contains only one word, the reply does not have to be enclosed in apostrophes.

SNDSRVRQS (Send Service Request) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 If ACTION(*OPENED) is specified, RMTCPNAME(*IBMSRV) cannot be specified.

2 The *SELECT parameter is not valid in batch mode.

Purpose

The Send Service Request (SNDSRVRQS) command establishes a communications session and sends problem information to the service support system.

Restriction: This command is shipped with public *EXCLUDE authority and the QSRV and QSRVBAS user profiles have private authorities to use the command.

Required Parameters

ACTION

Specifies the type of requests processed during this session.

***PREPARED:** All records in the problem log with a status of PREPARED are sent to the remote service support system.

***OPENED:** All records in the problem log with a status of OPENED are sent to the remote service support system.

***TEST:** A test is done on the remote support function and the communications link to it.

Optional Parameters

RMTCPNAME

Specifies the service provider to whom the service request is sent. When ACTION(*PREPARED) is specified, only the problem log entries that have defined destinations are processed.

***IBMSRV:** The service request is sent to IBM service support.

***SELECT:** A list of service providers is shown so the user can select the destination for the service request.

remote-control-point-name: Specify the remote control point name of the service provider to whom the request is sent.

RMTNETID

Specifies the remote name of the service provider's network.

***NETATR:** The service provider is in the local network.

remote-network-identifier: Specify the network name of the service provider to whom the request is being sent.

Examples

Example 1

```
SNDSRVRQS  ACTION(*PREPARED)
```

This command establishes the communications link to the IBM service support system and sends all records in the problem log with the status prepared. The result of each problem log entry reported may be one of the following:

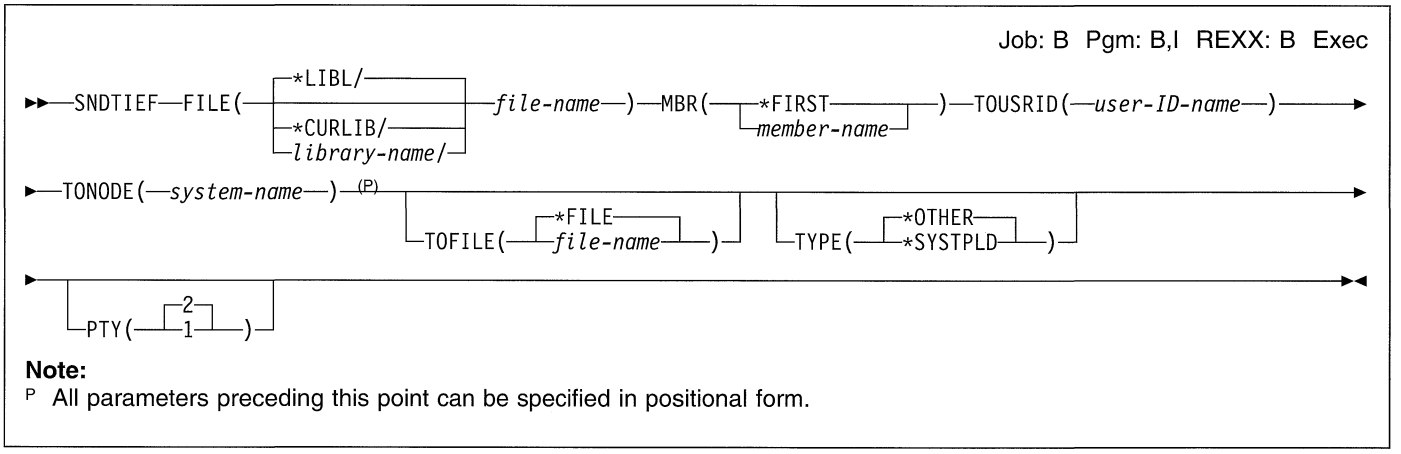
- PTFs sent to the system
- PTFs ordered from the code distribution center
- CE contacted automatically
- Service support center contacted automatically. The service support center representative will call you.
- Parts list

Example 2

```
SNDSRVRQS  ACTION(*OPENED)  RMTCPNAME(*SELECT)
```

This command allows the user to select a service provider from a list. The service provider will receive all records in the problem log with an opened status.

SNDTIEF (Send Technical Information Exchange File) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Send Technical Information Exchange File (SNDTIEF) command sends a specified file to the remote support network.

Required Parameters

FILE

Specifies the physical file to be sent to the remote support network.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the physical file.

MBR

Specifies the name of the database file member transmitted to the remote support network.

The possible member names are:

***FIRST:** The library list is used to locate the database file.

member-name: Specify the name of the library where the database file is located.

TOUSRID

Specifies the ID of the user who receives the file.

TONODE

Specifies the name of the system that receives the file.

Optional Parameters

TOFILE

Specifies the name of the file on the receiving system.

***FILE:** The name of the database file on the receiving system is the same as that specified on the FILE parameter.

file-name: Specify the name of the database file on the receiving system.

TYPE

Specifies the type of contents of the file.

***OTHER:** The content of the file is not specified.

***SYSTPLD:** The file contains system configuration descriptions.

PTY

Specifies the transmission priority that is assigned to the file.

2: The file has normal priority.

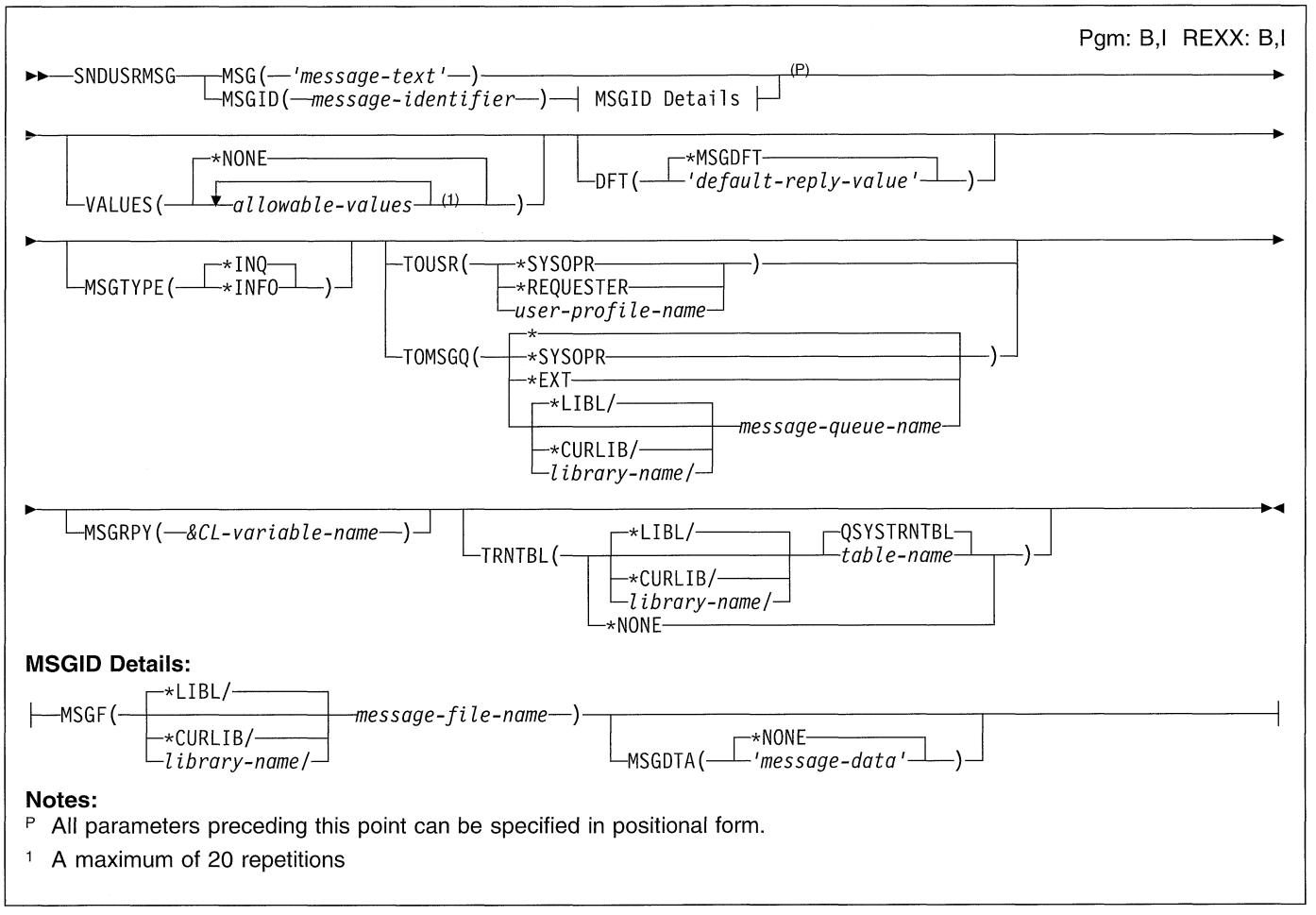
1: The file has the highest priority.

Example

```
SNDTIEF  FILE(QSYS/MYTOPO)  TOUSRID(INFSERV)
  TONODE(INFTIE)  TOFILE(ACMETOPO)
```

This command sends a file named MYTOPO from library QSYS to TIE. It is held in a mailbox for user INFSERV on system INFTIE. When it is received by the user, it is named ACMETOPO.

SNDUSRMSG (Send User Message) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 A maximum of 20 repetitions

Purpose

The Send User Message (SNDUSRMSG) command is used by a program to send a message to a message queue and, optionally, to receive a reply to that message. The message sent using this command can be either an impromptu message or a predefined message and it can be sent to a display station user either in an interactive job queue or in a specific message queue. For inquiry messages, a CL variable can be specified to receive the reply value, and the program using this command will wait for a response.

This command uses a combination of parameters available on the Send Program Message (SNDPGMMSG) and Receive Message (RCVMSG) commands to allow a program to send and receive messages by using a single command. Also, the SNDUSRMSG command provides validity checking and uppercase translation for replies to inquiry messages.

Restrictions:

1. This command is valid only in CL programs.
2. The SNDUSRMSG command allows a message of up to 512 characters of first-level message text to be sent.

However, if the message is sent to an external message queue (*EXT) in an interactive job, only 76 characters are shown on the Display Program Messages display. If the message is sent to a user's, work station's, or system operator's message queue, the Display Message (DSPMSG) command allows all 512 characters to be displayed.

Required Parameters

MSG

Specifies the message text that is sent. A maximum of 512 characters can be specified. The string must be enclosed in apostrophes if special characters (including blanks) are used. If a value is specified on this parameter, the MSGID, MSGF, and MSGDTA parameters cannot be specified.

MSGID

Specifies the message identifier of a predefined message sent by the program to a message queue. If a value is specified for the MSGID parameter, a value must also be specified for the MSGF parameter, while the MSG parameter cannot be specified.

MSGF

Specifies the qualified name of the message file that contains the predefined message (specified in the MSGID parameter) being sent. The MSGF parameter is valid only if a predefined message is being sent.

The name of the message file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-file-name: Specify the name of the message file to use.

Optional Parameters

MSGDTA

Specifies the character string or the CL variable that contains a character string used as the message data in the predefined message. A character string that is specified contains one or more substitution values that are used in place of the substitution variables that were defined in the message's text when the message was defined. The MSGDTA parameter is valid only if a predefined message is being sent. For more information, refer to the description of the MSGDTA parameter of the Send Program Message (SNDPGMMSG) command.

***NONE:** No message data is specified for the predefined message.

'message-data': Specify the character string that gives the substitution values in the specified predefined message that is sent, or specify the name of the variable that contains the character string.

VALUES

Specifies a list of valid replies to an inquiry message sent by this command. No more than 20 replies can be specified in the list. If the reply to the inquiry message does not match one of the specified values, an error message is sent to the reply's sender and the inquiry message is sent again. This value is valid only when a value is specified for the MSGRPY parameter.

Translation of the reply, if requested (on the TRNTBL parameter), will occur before verification of a match between the reply and the specified value. If TRNTBL(*LIBL/QSYSTRNTBL) is used to translate the reply values from lowercase to uppercase and lowercase characters still appear in a specified value, there is no match.

Variables can be used in the list of replies. This parameter is valid only if MSGTYPE(*INQ) is specified.

***NONE:** No predefined replies to inquiry messages are given. Any reply to an inquiry message is valid.

allowable-values: Specify no more than 20 values that are compared to replies received for inquiry messages sent by this command. The maximum length of each value is 32 characters.

DFT

Specifies the value used as the reply to an inquiry message (sent by this command) if the inquiry message is sent to a message queue that is in default delivery mode, or if for any other reason the default reply is sent. The value specified here as the default reply does not have to match an entry in the list of valid replies specified in the VALUES parameter. The default reply value is not translated to uppercase or validity-checked. This parameter is valid only if a value is specified for the MSGRPY parameter.

***MSGDFT:** The default value defined in the message description of the message ID (specified on the MSGID parameter) is used. If no message ID is specified, the default value is *N.

'default-reply-value': Specify the reply (enclosed in apostrophes) which is used as the default reply. If a predefined message is being sent, this default-reply-value overrides any default reply specified in the message description.

MSGTYPE

Specifies the type of message sent. Only inquiry and informational messages can be specified.

***INQ:** An inquiry message is sent and the message queue receiving the message must reply to it.

***INFO:** An informational message is sent.

TOUSR

Specifies that the message is sent to the message queue specified in the user profile for the user named on this parameter. This parameter cannot be used if TOMSGQ is specified.

***SYSOPR:** The message is sent to the system operator's message queue, QSYS/QSYSOPR.

***REQUESTER:** The message is sent to the user profile's message queue for interactive jobs or to the system operator's message queue (QSYS/QSYSOPR) for batch jobs.

user-profile-name: Specify the user profile name of the user to whom the message is sent.

TOMSGQ

Specifies the qualified name of the message queue to which the message is sent. This parameter cannot be used if TOUSR is specified.

***:** In an interactive job, the message is sent to the external message queue (*EXT). In a batch job, the message is sent to the system operator's (QSYS/QSYSOPR) message queue.

***SYSOPR:** The message is sent to the system operator's message queue, QSYS/QSYSOPR.

***EXT:** The message is sent to the job's external message queue. For batch job inquiry messages, the default reply is always received. Messages sent to this queue can be 512 characters in length, but only 76 characters are displayed on the Display Program Messages display.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue that is to receive the message being sent.

MSGRPY

Specifies the CL character variable (of up to 132 characters) that contains the reply received in response to an inquiry message. This parameter is valid only if MSGTYPE(*INQ) is specified. If the reply is longer than the variable, it is truncated; if it is shorter, it is padded to the right with blanks. This command will wait for a response to the inquiry message even if no value is coded for this parameter when MSGTYPE(*INQ) is specified. If no value is coded for this parameter when MSGTYPE(*INQ) is specified, the command awaits a response from the operator in reply to the inquiry message.

TRNTBL

Specifies, if the reply value is being translated, the qualified name of the translate table used.

The name of the translate table can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QSYSTRNTBL: The translate table named QSYSTRNTBL (found in library *LIBL) is used to translate the reply value.

The IBM-supplied translate table QSYSTRNTBL table translates, for the English language only, all lowercase characters in the range of X'81' to X'A9' to uppercase characters. All other characters are not translated.

To use a different translate table, use the Create Table (CRTTBL) command and specify that particular table for this parameter.

table-name: Specify the name of the translate table used to translate the message reply.

Other Single Values

***NONE:** The reply is not being translated.

Examples

Example 1: Message Requiring Specific Reply

```
SNDUSRMSG  MSG('Data has been verified.  Do you
  want to update the master files (Y or N)?')
  TOMSGQ(*) VALUES(Y N) DFT(N) MSGRPY(&REPLY)
```

This command sends an inquiry message to the display station operator (if it is used in an interactive job) or to the system operator (if it is used in a batch job). The valid replies are Y and N, and any other reply is rejected. The reply is returned in the variable &REPLY. The default translation table, QSYSTRNTBL, is used to translate the reply to uppercase characters.

Example 2: Message Requiring Any Reply

```
SNDUSRMSG  MSG('Enter any response to this
  message when ready to continue with updates.')
  TOMSGQ(WS01)
```

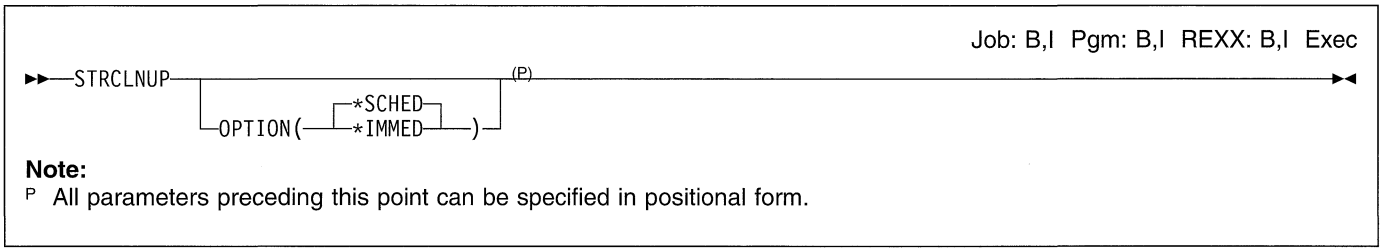
This command sends an inquiry message to a specific message queue. Any reply is valid. Because the purpose of this example is simply to wait, no CL variable is provided to receive the reply.

Example 3: Sending an Information Message

```
SNDUSRMSG  MSGID(USR0150)  TOUSR(FRED)
  MSGF(QGPL/USRMSGF)
  MSGDTA(&ACCTNO)  MSGTYPE(*INFO)
```

This command sends a predefined message as an information message to the message queue specified in the user profile of the specified user (FRED). The message data provided is included in the message.

STRCLNUP (Start Cleanup) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Start Cleanup (STRCLNUP) command checks to determine whether the cleanup operation is allowed. The cleanup operation is allowed if ALWCLNUP(*YES) is specified on the Change Cleanup (CHGCLNUP) command.

A batch job that controls the cleanup operation and the power on/off schedule is submitted to the job queue specified on the CHGCLNUP command. This cleanup control job submits individual batch jobs to the same job queue each day at the time specified on the STRTIME parameter of the CHGCLNUP command. These batch jobs do the actual cleanup of the objects specified on the CHGCLNUP command if cleanup is allowed.

More information is in the CHGCLNUP command description and the *Operator's Guide*.

Restriction: The user of this command must have *JOBCTL authority and must be authorized to the QPGMR user profile.

Optional Parameters

OPTION

Specifies when the cleanup operation is started.

***SCHED:** The cleanup operation is started as specified on the STRTIME parameter of the CHGCLNUP command or on the Change Cleanup Options display.

***IMMED:** The cleanup operation is started immediately.

Examples

Example 1: Starting Cleanup Operation as Scheduled

```
STRCLNUP
```

This command starts the cleanup operation as specified.

Example 2: Starting Cleanup Operation as Specified on CHGLNUP Command

```
STRCLNUP  OPTION(*SCHED)
```

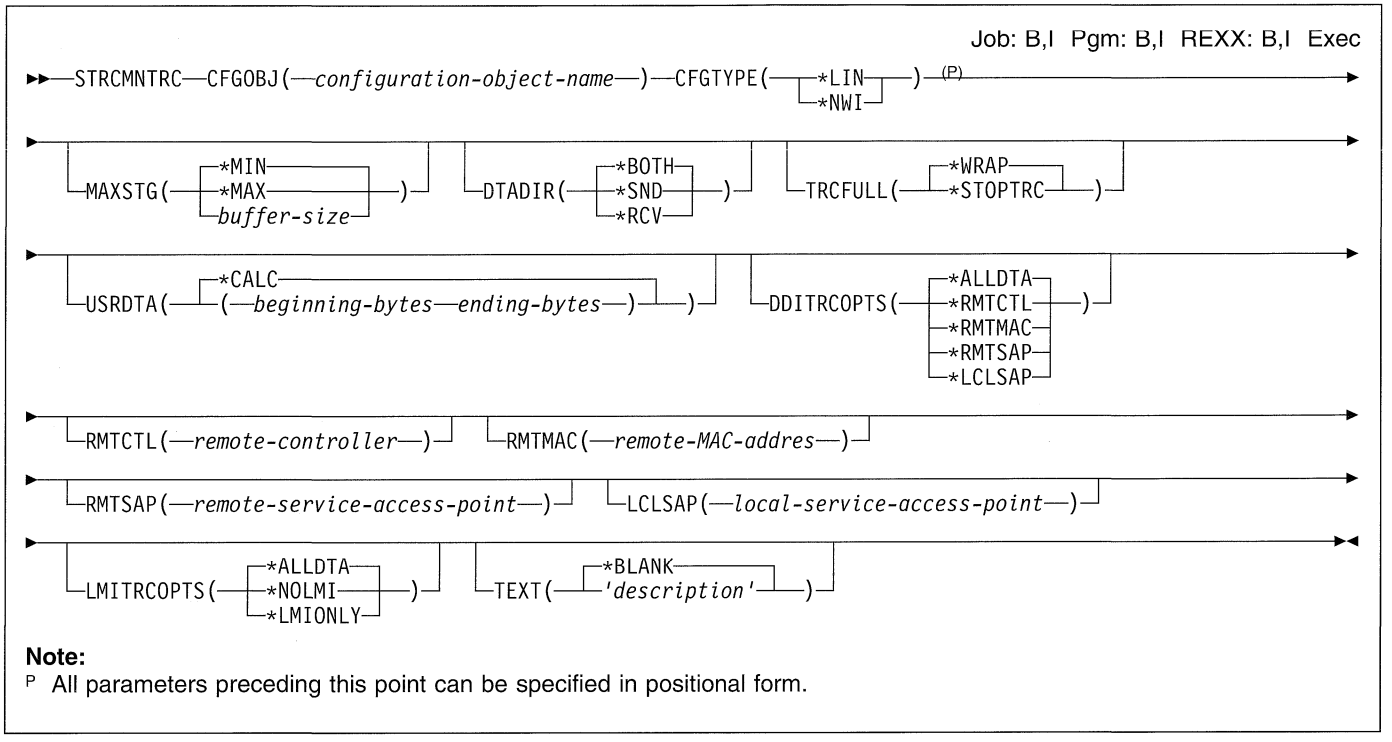
This command starts the cleanup operation as specified on the STRTIME parameter of the CHGCLNUP command or on the Change Cleanup Options display.

Example 3: Starting Cleanup Operation Immediately

```
STRCLNUP  OPTION(*IMMED)
```

This command starts the cleanup operation immediately.

STRCMNTRC (Start Communications Trace) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Start Communications Trace (STRCMNTRC) command initiates a communications trace for a specified line or network interface description. A communications trace continues until the End Communications Trace (ENDCMNTRC) command is run, a physical line problem causes the trace to end, or, if TRCFULL(*STOPTRC) is specified, the buffer becomes full.

Restrictions:

1. The user must have *USE authority to the line or network interface to be traced.
2. *SERVICE authority is required to use this command.
3. This command is shipped with public *EXCLUDE authority.
4. The following user profiles have authority to this command:
   - QSECOFR
   - QSRV

Required Parameters

CFGOBJ

Specifies the name of the configuration object to trace. The object is either a line description or a network interface description.

CFGTYPE

Specifies the type of configuration description to trace.

***LIN:** The type of configuration object is a line description.

***NWI:** The type of configuration object is a network interface description.

Optional Parameters

MAXSTG

Specifies the trace buffer size.

***MIN:** The minimum trace buffer size (128 kilobytes) is used.

***MAX:** The maximum trace buffer size (8192 kilobytes) is used.

buffer-size: Specify (in kilobytes) the trace buffer size. Valid values are 128, 256, 2048, 4096, 6144, 8192.

DTADIR

Specifies the communication data to trace.

***BOTH:** Data sent and received by the system is traced.

***SND:** Data sent by the system is traced.

***RCV:** Data received by the system is traced.

TRCFULL

Specifies the action the system takes when the trace buffer is full of data.

***WRAP:** The trace continues and overwrites the data in the buffer.

***STOPTRC:** The trace stops.

USRDTA
Specifies the amount of beginning and ending user data to trace.

***CALC:** The minimum amount of beginning and ending user data is traced.

Element 1: Beginning Bytes

beginning-bytes: Specify the amount (in bytes) of beginning user data to be traced.

Element 2: Ending Bytes

ending-bytes: Specify the amount (in bytes) of ending user data to be traced.

DDITRCOPTS
Specifies the type of data to be traced.

***ALLDTA:** All data is traced. No filtering is specified.

***RMTCTL:** The data traveling to and from a remote controller is traced.

***RMTMAC:** The data traveling to and from a remote medium access control (MAC) address is traced.

***RMTSAP:** The data traveling to and from a remote service access point (SAP) is traced.

***LCLSAP:** The data traveling to and from a local service access point (SAP) is traced.

RMTCTL
Specifies the remote controller receiving and sending the data to be traced.

RMTMAC
Specifies the remote medium access control address receiving and sending the data to be traced.

RMTSAP
Specifies the remote service access point receiving and sending the data to be traced.

LCLSAP
Specifies the local service access point receiving and sending the data to be traced.

LMITRCOPTS
Specifies the type of data to be placed in the trace buffer.

***ALLDTA:** All data, including the Local Management Interface (LMI), is placed in the trace buffer.

***NOLMI:** All data, except LMI data, is placed in the trace buffer.

***LMIONLY:** Only LMI data is placed in the trace buffer.

TEXT
Specifies text that briefly describes the trace. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: The text is used on spooled output and the System Service Tools (SST) Communications Trace screen displays.

***BLANK:** Text is not specified.

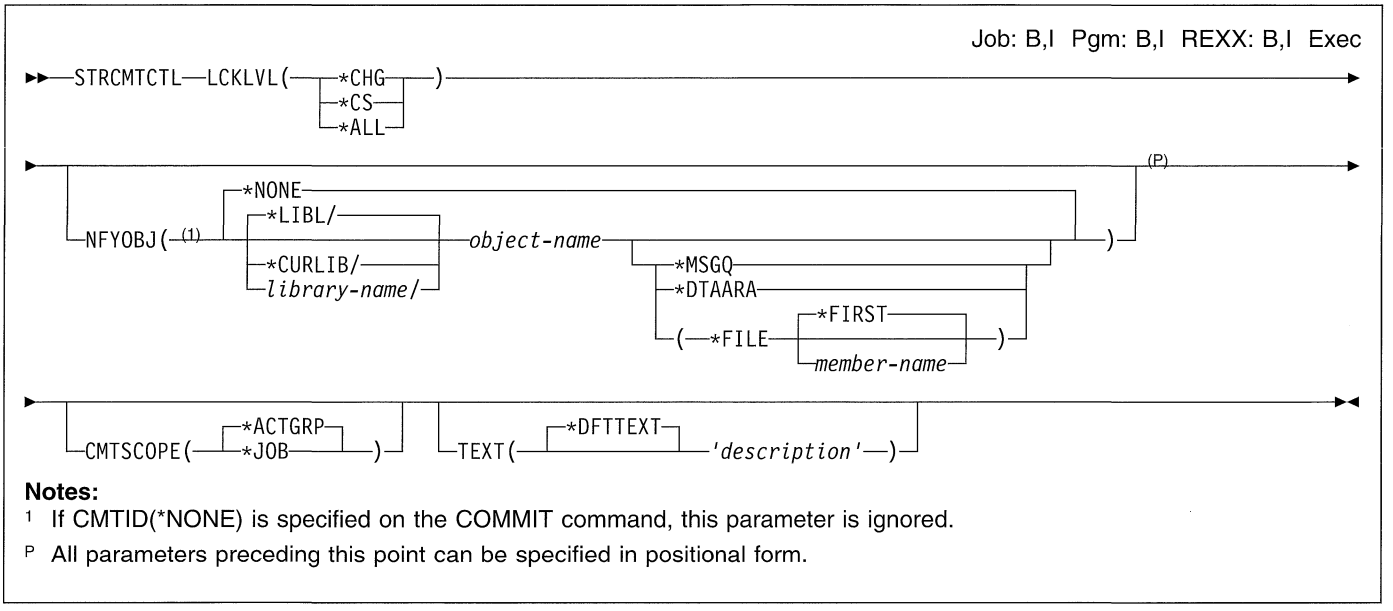
description: Specify up to 20 characters of text.

Example

```
STRCMNTRC  CFGOBJ(*QESLINE)  CFGTYPE(*LIN)
```

This command starts a communications trace of line description QESLINE.

STRCMTCTL (Start Commitment Control) Command



Notes:

1 If CMTID(*NONE) is specified on the COMMIT command, this parameter is ignored.

P All parameters preceding this point can be specified in positional form.

Purpose

The Start Commitment Control (STRCMTCTL) command is used to establish either a job level or activation group level commitment definition.

This command also specifies the level of record locking that occurs for the commitment definition to be started. Also, a notify object can be specified.

Before a commitment definition is established, the user must ensure that all database files that are to be opened under commitment control for a single commitment transaction are journaled to the same journal. If only the after images are being journaled, the system implicitly begins journaling both before and after images for the duration of the changes being made to files opened under this commitment definition.

More information on the use of journal management or the use of commitment control is in the *Advanced Backup and Recovery Guide*.

Restriction: The user must have object operational and add authority to the object named on the NFYOBJ parameter, if an object is specified.

Required Parameter

LCKLVL

Specifies the default level of record locking that occurs for the commitment definition to be started.

***CHG:** Every record read for update (for a file opened under commitment control) is locked. If a record is updated, added, or deleted, that record remains locked until the transaction is committed or rolled back. Records that are accessed for update but are released without being updated are unlocked.

***CS:** Every record accessed for files opened under commitment control is locked. Records that are not updated or deleted are locked only until a different record is accessed. Records that are updated, added, or deleted are locked until the transaction is committed or rolled back.

***ALL:** Every record accessed for files opened under commitment control is locked until the transaction is committed or rolled back.

Optional Parameters

NFYOBJ

Specifies the name and type of the object where notification is sent regarding the status of a transaction for a commitment definition. The commitment identifier of the last successful commit operation is sent to the notify object only for the following conditions:

- For a job level commitment definition, if any of the following are true:
  - A system failure occurs
  - The job ends with uncommitted changes
  - The job ends with a nonzero completion code
- For an activation group level commitment definition, if any of the following are true:
  - A system failure occurs
  - The job ends with uncommitted changes
  - The job ends with a nonzero completion code
  - The activation group ends abnormally
  - The activation group ends with uncommitted changes and the uncommitted changes are rolled back

For a system failure, the commitment identifier is placed in the notify object after the next successful initial program load (IPL). For a job that ends with uncommitted changes or with a nonzero completion code, the commitment identifier is placed in the notify object during end job processing. For an activation group that ends with uncommitted changes or ends abnormally, the commitment identifier is placed in the notify object during activation group end processing.

A commit identifier (specified on the CMTID parameter on the Commit (COMMIT) command) can be specified on each commit operation performed for a commitment definition. If more than one job is concurrently using commitment control or there is more than one commitment definition being used concurrently within a single job, then each commitment definition for each job should use a unique notify object or the specified commit identifier should contain unique text such that the text identifies a single commitment definition for a single job. If COMMIT CMTID(*NONE) is specified, this parameter is ignored.

***NONE:** No notification is sent after a system failure or during job end.

Element 1: Name of Object to Receive Notification

The name of the object can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

object-name: Specify the name of the object to receive notification of the last transaction that is successfully committed. The user must have the required authority for the object specified.

Element 2: Location of Text

***MSGQ:** The text identifying the last commitment boundary is placed on the specified message queue.

***DTAARA:** The text identifying the last commitment boundary is placed in the specified data area. The data area specified must be of type character, and unique to this job. The text is padded or truncated to fit the data area.

***FILE:** The text identifying the last commitment boundary is added to the specified physical file.

Element 3: Member to Receive Notification

This element is valid only when *FILE is specified.

***FIRST:** The first member of the physical file receives the notification.

member-name: Specify the name of the member of the physical file that receives the notification.

CMTSCOPE

Specifies the scope for the commitment definition to be started.

***ACTGRP:** An activation group level commitment definition is started for the activation group associated with the program issuing the command.

***JOB:** The job level commitment definition is started for the job.

TEXT

Specifies text that briefly describes the commitment definition to be started. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***DFTTEXT:** The system is to provide a default text description for the commitment definition.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

Examples

Example 1: Defining Activation Group Level Commitment

```
STRCMTCTL  LCKLVL(*CHG)
  CMTSCOPE(*ACTGRP)
  TEXT('Blue Commit Group')
```

This command described by the user as the Blue Commit Group starts the activation group level commitment for the activation group associated with the program issuing the command.

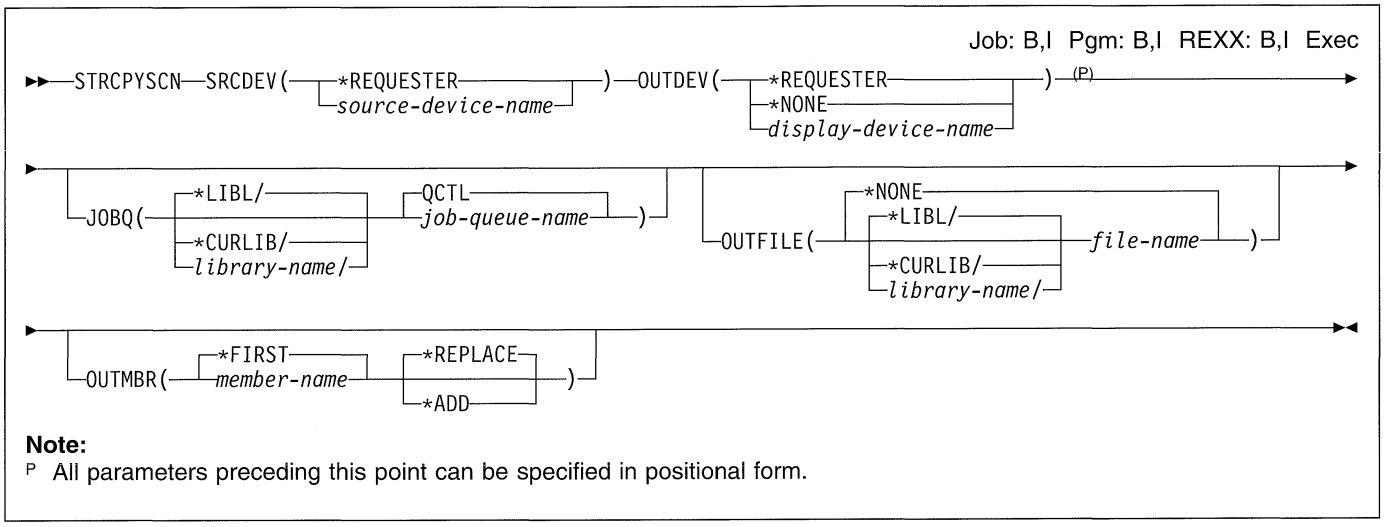
Only records that are updated, inserted, or deleted are locked until the transaction is ended by a commit or rollback operation. No identification for the commitment boundary is sent after the initial program load (IPL) following an abnormal system end, after an abnormal end to an activation group for the job, or when the job or activation group ends either with uncommitted changes or with a nonzero completion code.

Example 2: Defining Job Level Commitment

```
STRCMTCTL  LCKLVL(*ALL)
  NFYOBJ(RCVLIB/MYFILE *FILE IDSAVE)
  CMTSCOPE(*JOB)
```

This command starts the job level commitment definition. All records accessed in files opened under commitment control are locked until the commitment transaction is ended by a commit or rollback operation. If a commitment transaction ends in a manner that a notify object is to be updated with the commitment identifier of the last successful commit operation, the notify object to be updated is member IDSAVE of file MYFILE in the library RCVLIB.

STRCPYSCN (Start Copy Screen) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Start Copy Screen (STRCPYSCN) command allows the user to copy the displays of another display station to observe what is happening and diagnose problems.

If the STRCPYSCN command is used to copy displays from a source device that has the wide-display feature to an output device with a regular-width display, the command is accepted, but wide-display images are not shown and an informational message is sent to the target work station indicating that the display was not shown.

If the STRCPYSCN command is used to copy displays from a source device that supports graphic DBCS characters, the command is accepted and character information is shown, but graphic DBCS characters appear as single byte. No message is sent.

If the output device is not the requesting device, then the output device cannot be signed on. If the output device is signed on, a message is sent to the requester indicating that the device is not available for copying. If the source device is signed off after display copy has begun, the function automatically ends.

Note: The copy display function can be ended by either the source or target device.

Restrictions:

1. Permission must be given from the user of the source work station.
2. When a request is made to begin display image copying, a break message is sent to the user of the source work station to inform the user that the displays are going to be copied. The user must reply to this message before any displays are copied.
3. *REQUESTER is not valid for the SRCDEV or OUTDEV parameters when the command is submitted to batch.

Required Parameters

SRCDEV

Specifies the display station that is used as the source for the display images being copied.

***REQUESTER:** The displays are copied from the display station that issued this command.

source-device-name: Specify the name of the display station name (other than the *REQUESTER) whose displays are being copied.

OUTDEV

Specifies the output device for the copying process.

***REQUESTER:** The displays are copied to the work station from which this command is issued. If *REQUESTER is specified on the SRCDEV parameter, *REQUESTER cannot be specified on the OUTDEV parameter.

***NONE:** The copied displays are not sent to a display station. If OUTDEV is *NONE, then OUTFILE must be specified.

display-device-name: Specify the name of the display station (other than the *REQUESTER) that shows the copied displays.

Optional Parameters

JOBQ

Specifies the job queue that is used to submit the job that shows the displays from the source device on the target device when the requesting device is not the target device. When OUTDEV(*REQUESTER) is specified, this parameter is ignored, since it defaults to the values for the target display station and then a submit job command is not necessary.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

The library list is used to locate the job queue.

QCTL: The default job queue, QCTL, is used.

job-queue-name: Specify the name of the job queue where the job of processing the copied displays is submitted.

OUTFILE

Specifies the qualified name of the database file where the copied displays are directed. If the database file does not exist, this command creates a database file in the specified library. Displays can be copied to a database file instead of, or in addition to, being shown on another work station, thereby keeping a record of the displays that were copied and shown.

Note: A library name of QTEMP cannot be used.

***NONE:** The output is not directed to a database file.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the database file that receives the output of the display.

If the user qualifies the database file name with *LIBL, but the system cannot find the file, the file is created in the QGPL library. Or, if a library name is specified, the file is created in the specified library.

OUTMBR

Specifies the name of the database file member to which the output is directed.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The system clears the existing member and adds the new records.

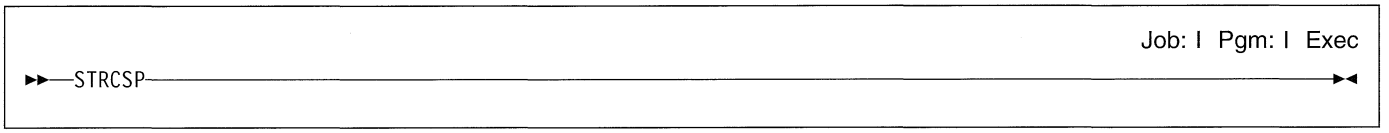
***ADD:** The system adds the new records to the end of the existing records.

Example

```
STRCPYSCN  SRCDEV(WS2)  OUTDEV(*REQUESTER)
```

This command sends an inquiry message to the user of work station, WS2. The message indicates that the display station displays are about to be copied to another display station. If the user of that display does not wish this to happen, then a cancel (C) reply prevents the operation from beginning. To allow the operation to begin, the user responds with a go (G) to the message.

STRCSP (Start CSP/AE Utilities) Command



Purpose

The Start CSP/AE Utilities (STRCSP) command calls the Cross System Product/Application Execution (CSP/AE) utilities program. A menu is shown that allows you to create objects, create messages, and work with CSP/AE objects.

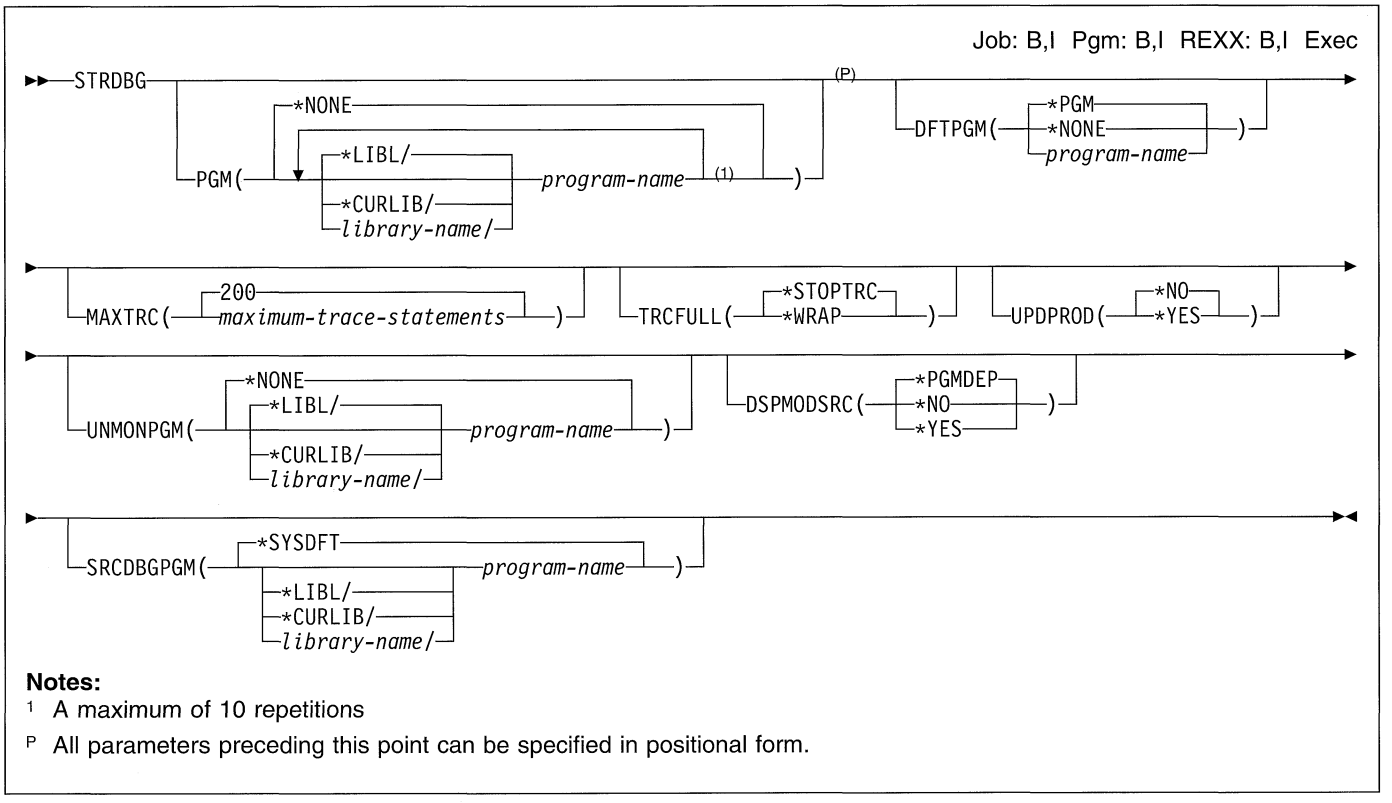
There are no parameters for this command.

Example

```
STRCSP
```

This command starts the CSP/AE utilities menu.

STRDBG (Start Debug) Command



Notes:

1 A maximum of 10 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Start Debug (STRDBG) command allows the user to put a job into debug mode. Also, this command provides the option to add as many as 10 programs to debug mode. It also specifies certain attributes of the debugging session. For example, it can specify whether database files in production libraries can be updated while in debug mode.

The Change Debug (CHGDBG) command can be used later in the job to change the attributes of the debug mode. Also, programs can be added to or removed from the debugging session if they are specified in the Add Program (ADDPGM) or Remove Program (RMVPGM) commands.

When one job is servicing another job, and the STRDBG command is entered, all debug commands are valid for the job being serviced. To service another job, see the STRSRVJOB (Start Service Job) command. More information about debugging one job from another job is in the *CL Programmer's Guide*.

Restrictions:

1. This command is shipped with public *EXCLUDE authority, and the QPGMR, QSRV, and QSRVBAS user profiles have private authorities to use the command.
2. The user must have either *CHANGE authority to the program, or *USE authority to the program and *SERVICE special authority.
3. This command is not valid in debug mode. To end debug mode, see the ENDDBG (End Debug) command.
4. This command cannot be used if the user is servicing another job, and that job is being held, suspended, or ended.

Optional Parameters

PGM

Specifies the qualified names of up to 10 programs to debug in the job. Before a program can be debugged, its name must be specified on this parameter or in the ADDPGM command.

***NONE:** No program names are specified at the start of the debugging session. The ADDPGM command can be used to add programs later.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the names of up to 10 programs to debug. The user cannot debug two programs that have the same name at the same time.

DFTPGM

Specifies the name of the program to use as the default program during debug mode. The program specified here is used as the default program for any of the other debug commands that specify *DFTPGM on their PGM parameter. That is, if a default program was previously specified, this parameter can change it.

***PGM:** The program named in the PGM parameter of *this* command is the default program for the job's debugging environment. If PGM has more than one program name specified, the first program named in the list is the default program. If PGM(*NONE) is specified or is the default, DFTPGM(*NONE) is also assumed when *PGM is specified here.

***NONE:** No program is specified as the default program; if a program was specified as a default program, it is no longer the default program. If the job has no default program, *DFTPGM cannot be specified on the PGM parameter of any other debug commands.

program-name: Specify the name of the program to use as the default program during debug mode. The same qualified name must also be specified in the PGM parameter of this command. A bound program cannot be specified on this parameter.

MAXTRC

Specifies the maximum number of trace statements that the system puts into the job's trace file before either stopping, tracing, or wrapping around (overlaying) on the trace file. When the trace file contains the maximum number specified, the system performs the actions specified in the TRCFULL parameter.

Note: Instruction stepping can be performed on a program being debugged in an interactive environment by setting the maximum number of trace statements to 1 and specifying *STOPTRC on the TRCFULL parameter.

200: Up to 200 trace statements can be put into the trace file before tracing is ended or wrapping on the file occurs.

maximum-trace-statements: Specify the maximum number of trace statements that can be in the trace file.

TRCFULL

Specifies what happens when the job's trace file is full; that is, it contains the maximum number of trace statements specified by the MAXTRC parameter.

***STOPTRC:** In a batch environment, tracing stops but the program continues processing. In an interactive environment, control is given to the user when a breakpoint occurs. If the user continues processing, a breakpoint occurs before processing each subsequent statement within the range of statements being traced, and the trace file is extended to contain the new entry.

***WRAP:** The trace file is overlaid with new trace statements as they occur, wrapping from the beginning of the file. The program completes processing without sending a message to indicate that wrapping has occurred. The trace file never has more than the maximum specified statements, and they are the most recently recorded statements.

UPDPROD

Specifies whether or not database files in a production library can be opened for updating records, or for adding new records, while the job is in debug mode. If not, the files must be copied into a test library before trying to run a program that uses the files.

***NO:** Database files in production libraries cannot be updated while the job is in debug mode. Database files can be opened for reading only. This protects database files from unwanted updates while a program is being debugged. The exception to this is starting debug mode after a production library is already opened. This is possible if the file is opened with SHARE(*YES) in a different program first. To prevent this, start debug mode before any files are opened, or use SHARE(*NO) on the Override Database File (OVRDBF) command for all files used in the programs being debugged.

***YES:** Database files in production libraries can be updated while the job is in debug mode.

UNMONPGM

Specifies the qualified name of the user-supplied program called when a message that is not monitored occurs in the job being debugged. When the program specified is called, it is passed parameters that identify the program name, the recursion level, the high-level language statement identifier, the machine instruction number at which the breakpoint occurred, the message that was not monitored, the message data, the length of the message data, and the message reference key. The passed parameters have the following formats:

1. Program name (10 bytes). Specifies the name of the program in which the breakpoint was reached.
2. Recursion level (5 bytes). Specifies the recursion level number of the program in which the breakpoint was reached. This value is a 1- to 5-digit number padded on the right with blanks.
3. Statement Identifier (10 bytes). Specifies the high-level language program statement identifier that was reached. This is the statement identifier specified on the Add Breakpoint (ADDBKP) command. If a machine instruction number is used to specify the breakpoint, this parameter contains a slash (/) followed by a 4-digit hexadecimal machine instruction number.
4. Instruction number (5 bytes). Specifies the machine instruction number that corresponds to the high-level language statement at which the breakpoint was reached. No slash appears in front of the machine instruction number. The value consists of 1 to 4

hexadecimal characters representing the MI instruction number, followed by one or more blanks. If a machine instruction number is passed on the third parameter, the numbers in the third and fourth parameters are the same.

5. Message ID (7 bytes). Specifies the ID of the message that was not monitored.
6. Message data (256 bytes). Specifies the first 256 bytes of message data sent with the message not monitored.
7. Message data length (5 bytes). Specifies the length of the message data sent with the message not monitored.
8. Message MRK (4 bytes). Specifies the message reference key (MRK) of the message not monitored.

All the parameter values are left-adjusted and padded on the right with blanks. When control returns to the program with the message that was not monitored, processing continues.

***NONE:** No program is called when a message that is not monitored occurs.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of the user-supplied program called when a message that is not monitored occurs in the job being debugged. After the program runs, control is returned to the interrupted program.

DSPMODSRC

Specifies whether the first display of the source debug program is shown when this command is processed and no errors occur.

***PGMDEP:** The showing of the source debug program display is dependent on the first program specified on the PGM parameter. If the first program is a bound program, the display is shown. If the first program is not a bound program, the display is not shown.

***NO:** The first display of the source debug program is not shown.

***YES:** The first display of the source debug program is shown.

SRCDBGPGM

Specifies the source debug program to be used. See the *System Programmer's Interface Reference* for an explanation of this parameter.

***SYSDFT:** The system source debug program is used.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

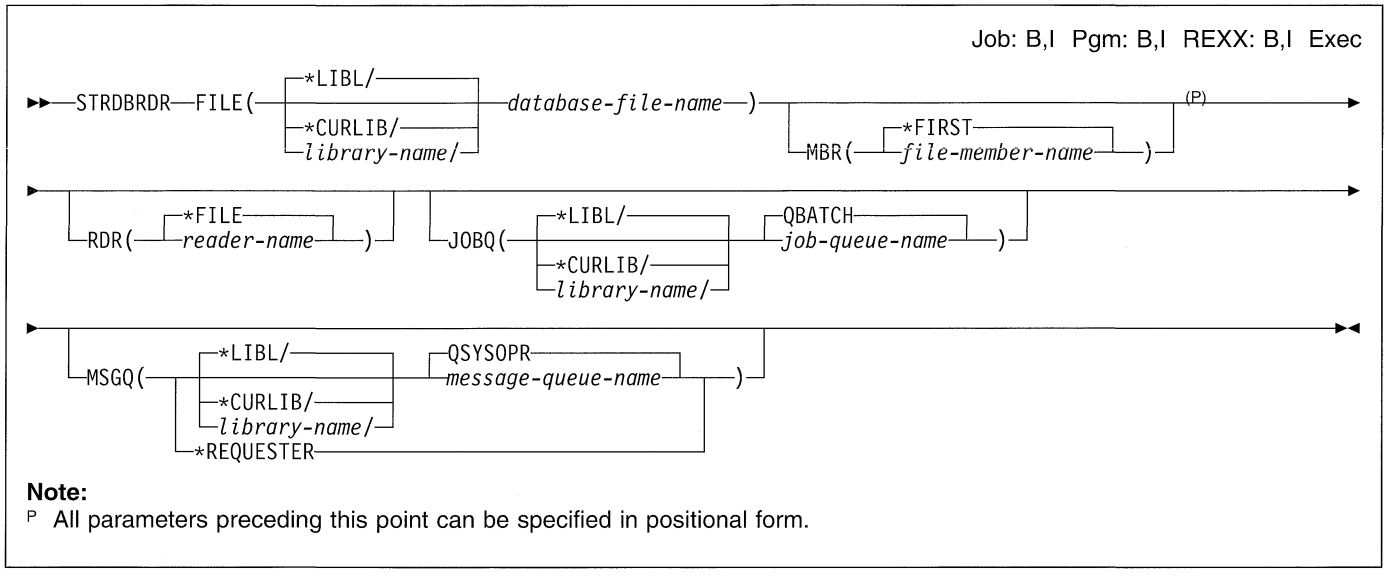
program-name: Specify the name of the program to be used to debug programs.

Example

```
STRDBG  PGM(TESTLIB/PAYROLL)
```

This command starts debug mode to debug the program PAYROLL, which is in the test library TESTLIB. If tracing is used, up to 200 trace statements can be stored in the trace before tracing stops. If program PAYROLL is a bound program, the Display Module Source display will be shown, giving the source for the module that contains the program entry point. Any database files updated by the PAYROLL program must be in a test library.

STRDBRDR (Start Database Reader) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Start Database Reader (STRDBRDR) command starts a spooled reader to a database file. The reader reads a batch input stream from the database file and places the jobs onto one or more job queues. This command specifies the name of the database file and member from which the input stream is read, the name of the reader, and the names of the job queue and message queue that are used.

More than one reader can be active at the same time (as determined by the spooled subsystem description). Each database reader must have a unique reader name, and the specified file or member must be available. The reader can also be held or canceled if the Hold Reader (HLDRDR) or End Reader (ENDRDR) command is used.

Because each reader runs independent of the job that started it, users can continue doing other work on the system after they have started a reader. The reader is owned by the user who issues the STRDBRDR command.

Restriction: The specified database file either must consist of single-field records and must have an arrival sequence access path, or it must be a standard database source file.

Required Parameters

FILE

Specifies the qualified name of the database file from which the input stream is to be read. The file must be available for allocation to the spooled reader before the reader can be started.

| The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file.

Optional Parameters

MBR

Specifies the name of the member in the specified file that contains the input stream that is read by the reader.

***FIRST:** The first member in the database file is used.

file-member-name: Specify the name of the member that contains the input stream that is read by the reader.

RDR

Specifies the name of the spooled reader being started. Each reader name must be unique.

***FILE:** The name of the reader is the same as that of the database file specified by the FILE parameter.

reader-name: Specify the name by which the reader being started is identified.

JOBQ

Specifies the qualified name of the job queue on which the spooled reader places jobs. An entry is placed on this job queue for each job in the input stream that specifies JOBQ(*RDR) on its Batch Job (BCHJOB) command. If *RDR is not specified on the BCHJOB command, the job queue specified in the BCHJOB command or in the job description is used. Each job in the input stream can have a different job queue.

Restriction: If the user starting the reader is not authorized for the job queue used by the reader for a job in the input stream, an error message is sent to the message queue specified by the MSGQ parameter and the job does not run.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QBATCH: The job entry is placed on the system-supplied QBATCH job queue. QBATCH is the default if JOBQ(*RDR) is specified on the BCHJOB command.

job-queue-name: Specify the qualified name of the job queue to which each job read by this reader is sent if JOBQ(*RDR) is specified on its BCHJOB command.

MSGQ

Specifies the qualified name of the message queue to which the messages created by the reader are sent.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QSYSOPR: The messages are sent to the system operator's message queue, QSYSOPR.

message-queue-name: Specify the name of the message queue to which messages created by the reader are sent.

Other Single Values

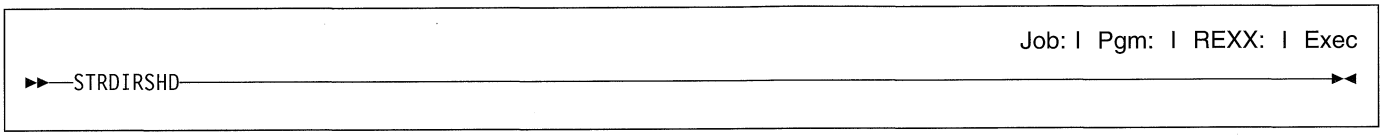
***REQUESTER:** The messages are sent to the message queue of the user who started the process. This value is not valid for batch jobs.

Example

```
STRDBRDR  FILE(QGPL/BILLING)
```

This command starts a spooled reader that reads its input from the database file named BILLING, which is in the QGPL library. The reader name is also BILLING because the RDR parameter was not specified. The first member in the BILLING file contains the input stream to be processed. The default job queue QBATCH and the system-supplied system operator's message queue QSYSOPR are used by the database reader.

STRDIRSHD (Start Directory Shadowing) Command



Purpose

The Start Directory Shadowing (STRDIRSHD) command submits a job to start the directory shadowing environment in the system work subsystem (QSYSWRK). The system administrator can use this command to restart the directory shadowing environment if it is not already active. Only one active directory shadowing environment per system is allowed. If the directory shadowing environment is already active, a warning message is issued.

The system work subsystem (QSYSWRK) automatically starts the directory shadowing environment as a prestart job when the subsystem is started.

To ensure the job submitted with this command is successful, use the Work with Jobs (WRKJOB) command using the job number returned in the message after issuing the STRDIRSHD command.

There are no parameters for this command.

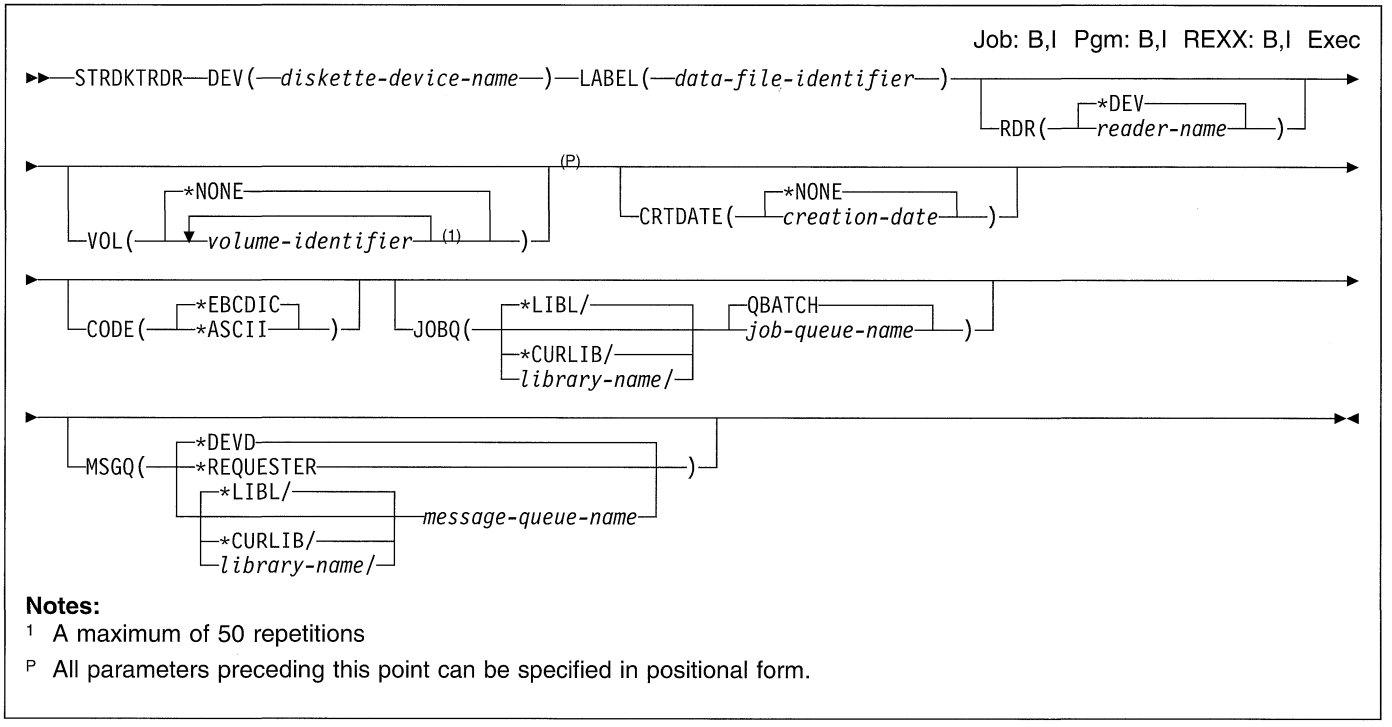
Restriction: You must have job control (*JOBCTL) authority to use this command.

Example

```
STRDIRSHD
```

This command submits a job to start the directory shadowing environment in the system work subsystem.

STRDKTRDR (Start Diskette Reader) Command



Notes:

1 A maximum of 50 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Start Diskette Reader (STRDKTRDR) command starts a spooled reader to the specified diskette device to read a batch input stream and place it on the appropriate job queue. This command specifies the name of the diskette device from which the input stream is read; the volume and data file that the input stream is on; and the names of the job queue and message queue to be used.

On a system job, more than one reader can be active at the same time (as determined by the spooled subsystem description). Each reader must have a unique reader name and have its own input device assigned to it. A reader that has been started can be actively reading input or waiting for device input. The reader can also be held or ended if the Hold Reader (HLDRDR) or End Reader (ENDRDR) command is used. The reader is ended at end-of-file.

Because each reader runs independent of the job that started it, users can continue doing other work on the system after they have started a reader. The reader is owned by the user who issues the STRDKTRDR command.

Restriction: This command cannot be used to read data files of diskettes that are in the save/restore format.

Required Parameters

DEV

Specifies the name of the diskette device that is used to read and enter an input stream into a job.

LABEL

Specifies the data file identifier of the file that is processed as an input stream. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Optional Parameters

RDR

Specifies the name of the spooled reader that is started. Each reader name must be unique.

***DEV:** The name of the reader is the same as that of the diskette device specified in the DEV parameter.

reader-name: Specify the name by which the reader being started is identified.

VOL

Specifies one or more volume identifiers used by the file. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***NONE:** No volume identifier is specified. Use the name of the current volume.

volume-identifier: Specify the identifiers of one or more volumes in the order that they are put on the device and read. Each identifier can have 6 alphanumeric characters or less. A blank is used as a separator character when listing multiple volumes.

CRTDATE

Specifies the creation date of the object.

Note: The creation date should not be specified if the

date is not checked against the diskette date by the system. If the date written on the diskette containing the data file does not match the date specified on this parameter, an error message is sent to the message queue specified on the MSGQ parameter.

***NONE:** The creation date is not specified and no check is made.

creation-date: Specify the creation date of the data file that is read. The date should be in the format specified by the QDATFMT system value.

CODE

Specifies the character code used. The code can be either extended binary-coded decimal interchange code (*EBCDIC) or the American National Standard Code for Information Interchange (*ASCII).

***EBCDIC:** The extended binary-coded decimal interchange code (EBCDIC) character set code is used.

***ASCII:** The ASCII character set code is used.

JOBQ

Specifies the job queue on which the job entries are placed. A job entry is placed on this queue for each job in the input stream that has JOBQ(*RDR) specified on the Batch Job (BCHJOB) command. If *RDR is not specified on the BCHJOB command, the job queue specified on the BCHJOB command or in the job description is used. (The job queue for each job in the input stream can be different.) This parameter is valid only if ACTION(*SUBMIT) is specified on this command, in the existing network job entry, or in a subsequent Change Network Job Entry (CHGNETJOBE) command.

Note: If both the user identified in the job description of the job being read and the user processing the Start Diskette Reader (STRDKTRDR) command are not authorized to the job queue on which the job should be placed, the job ends and a diagnostic message is placed in the job log. The input stream, continues to be processed, starting with the next job. If either user is authorized to the job queue, the job runs without error.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QBATCH: The job entry is placed on the QBATCH job queue.

job-queue-name: Specify the name of the job queue to which each job read by this reader is sent if JOBQ(*RDR) is specified on its JOB command.

MSGQ

Specifies the qualified name of the message queue to which messages are sent.

***DEVD:** The messages are sent to the message queue specified in the device description of the device named in the DEV parameter.

***REQUESTER:** The messages are sent to the message queue of the user who started the process. This value is not valid for batch jobs.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

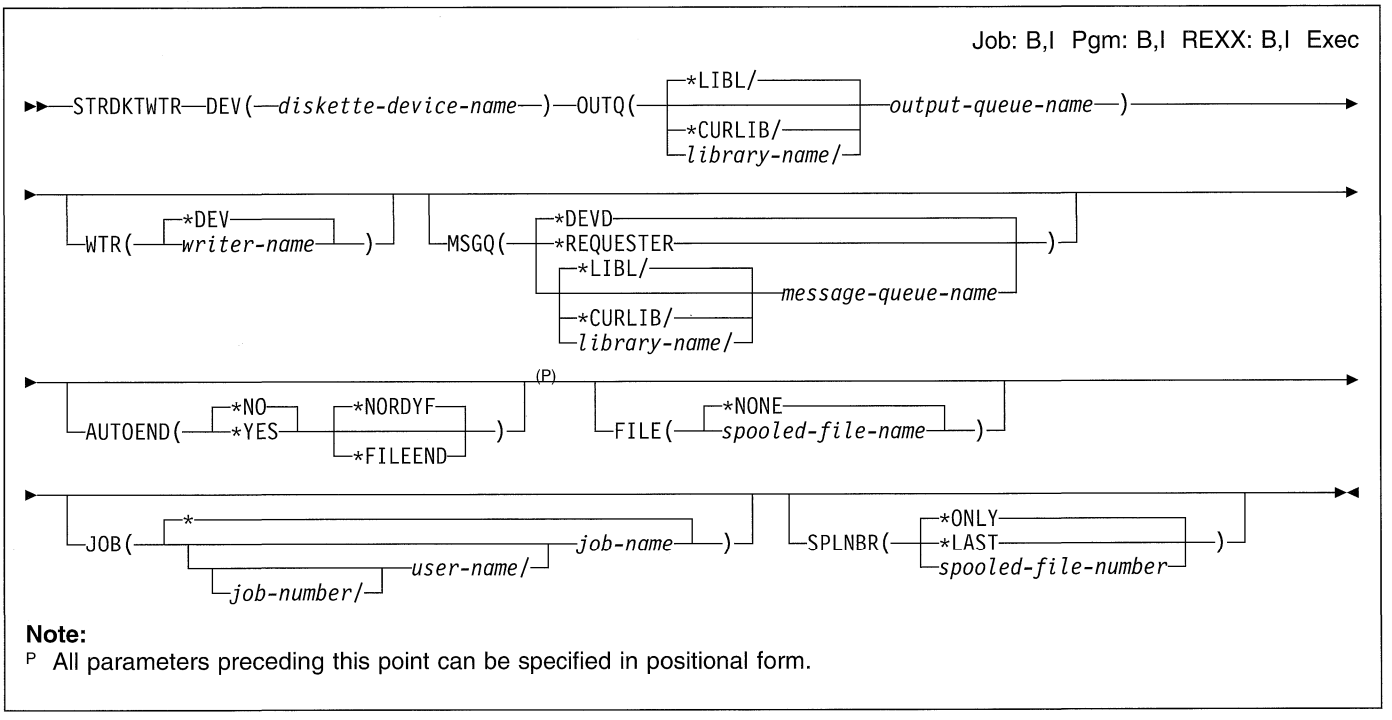
message-queue-name: Specify the name of the message queue to which the diskette reader messages created by the reader are sent.

Example

```
STRDKTRDR  DEV(QDKT)  LABEL(OCT24)  VOL(SALES)
```

This command starts the spooled reader named QDKT, which reads diskette input from the device QDKT. Because *DEV was the default on the unspecified RDR parameter, the device name QDKT is also used as the reader name. The reader reads its input from the data file named OCT24 whose volume identifiers must be SALES. The default job queue QBATCH and the message queue QSYSOPR are used by the diskette reader.

STRDKTWTR (Start Diskette Writer) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Start Diskette Writer (STRDKTWTR) command starts a spooling writer to the specified diskette device. The writer, which is a system job, takes spooled files from an output queue and produces (writes) the output on the diskette device. This command specifies the names of the diskette device and the diskette writer and the names of the output and message queues to be used.

More than one writer can be active at the same time (as determined by the spooling subsystem description). Each writer must have a unique writer name, its own output queue, and its own device. A writer that has been started can be actively writing output or waiting for a file to be put on the output queue. Optionally, the writer can be automatically stopped when it has processed all the files on the output queue. The writer can also be held or stopped if the Hold Writer (HLDWTR) or End Writer (ENDWTR) command is used.

Because each writer runs independent of the job that started it, users can continue doing other work on the system after they have started a writer.

Required Parameters

DEV

Specifies the name of the diskette device to which the spooled file is written.

OUTQ

Specifies the qualified name of the output queue.

The name of the output queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

output-queue-name: Specify the name of the output queue.

Optional Parameters

WTR

Specifies the name of the writer being started. Each writer name must be unique.

***DEV:** The name of the writer is the same as that of the diskette device specified in the DEV parameter.

writer-name: Specify the name by which the writer being started is identified.

MSGQ

Specifies the qualified name of the message queue to which messages are sent.

***DEVD:** The messages are sent to the message queue specified in the device description of the device named in the DEV parameter.

***REQUESTER:** The messages are sent to the message queue of the user who started the process. This value is not valid for batch jobs.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue to which the diskette writer messages are sent.

AUTOEND

Specifies whether the writer stops automatically when the output queue has no more available entries for spooled files to be written to the diskette device.

Element 1: Stop Writer Option

***NO:** The writer does not end when the last available entry has been removed from the output queue; it waits for another spooled file entry to be put on the queue.

***YES:** The writer automatically ends after it has reached the state specified by the second part of the parameter (*NORDYF or *FILEEND).

Element 2: Conditions for Stopping Writer

***NORDYF:** The writer automatically ends when there are no ready files (all the available entries have been removed from the output file).

***FILEEND:** The writer stops after it has finished processing one spooled file.

FILE

Specifies the name of the first (or only) spooled file processed by the spooling writer and written to diskette. If several files are available on the output queue, the next file produced is the first one available with the highest priority.

***NONE:** No spooled file name is specified; the first spooled file that becomes available on the output queue is processed first.

spooled-file-name: Specify the name of the spooled file that is the first (or only) spooled file to be written to diskette.

JOB

Specifies the name of the job that created the spooled file being written to diskette. This parameter is valid only if a spooled file name is specified in the FILE parameter. If no job qualifier is given, all of the jobs currently in the system are searched for the simple job name.

A job identifier is a special value or a qualified name with up to three elements. For example:

```
*
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** The job that issued this STRDKTWTR command is the job that created the spooled file.

job-name: Specify the name of the job that created the spooled file.

user-name: Specify the name of the user of the job that created the spooled file.

job-number: Specify the number of the job that created the spooled file.

SPLNBR

Specifies the number of the spooled file that is processed first. This parameter is valid only if a spooled file name is specified in the FILE parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ONLY:** One spooled file from the job has the specified file name. The number of the spooled file is not necessary. If *ONLY is specified and more than one spooled file has the specified file name, a message is sent.

***LAST:** The spooled file with the highest number and the specified file name is used.

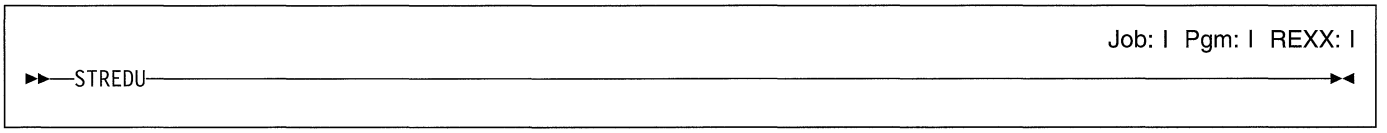
spooled-file-number: Specify the number of the specified file from the job on the specified output queue that is processed first.

Example

```
STRDKTWTR  DEV(QDKT)  OUTQ(QDKT)  AUTOEND(*YES)
```

This command starts a spooling writer to the diskette drive. The files written on the diskettes are on the IBM-supplied output queue QDKT. When all the files have been written (no more entries are on the QDKT output queue), the writer is automatically ended and the diskette drive is available for other uses.

STREDU (Start Education) Command



Purpose

The Start Education (STREDU) command starts the education session.

This command is shipped with public *EXCLUDE authority.

There are no parameters for this command.

Example

```
STREDU
```

This command shows the following menus:

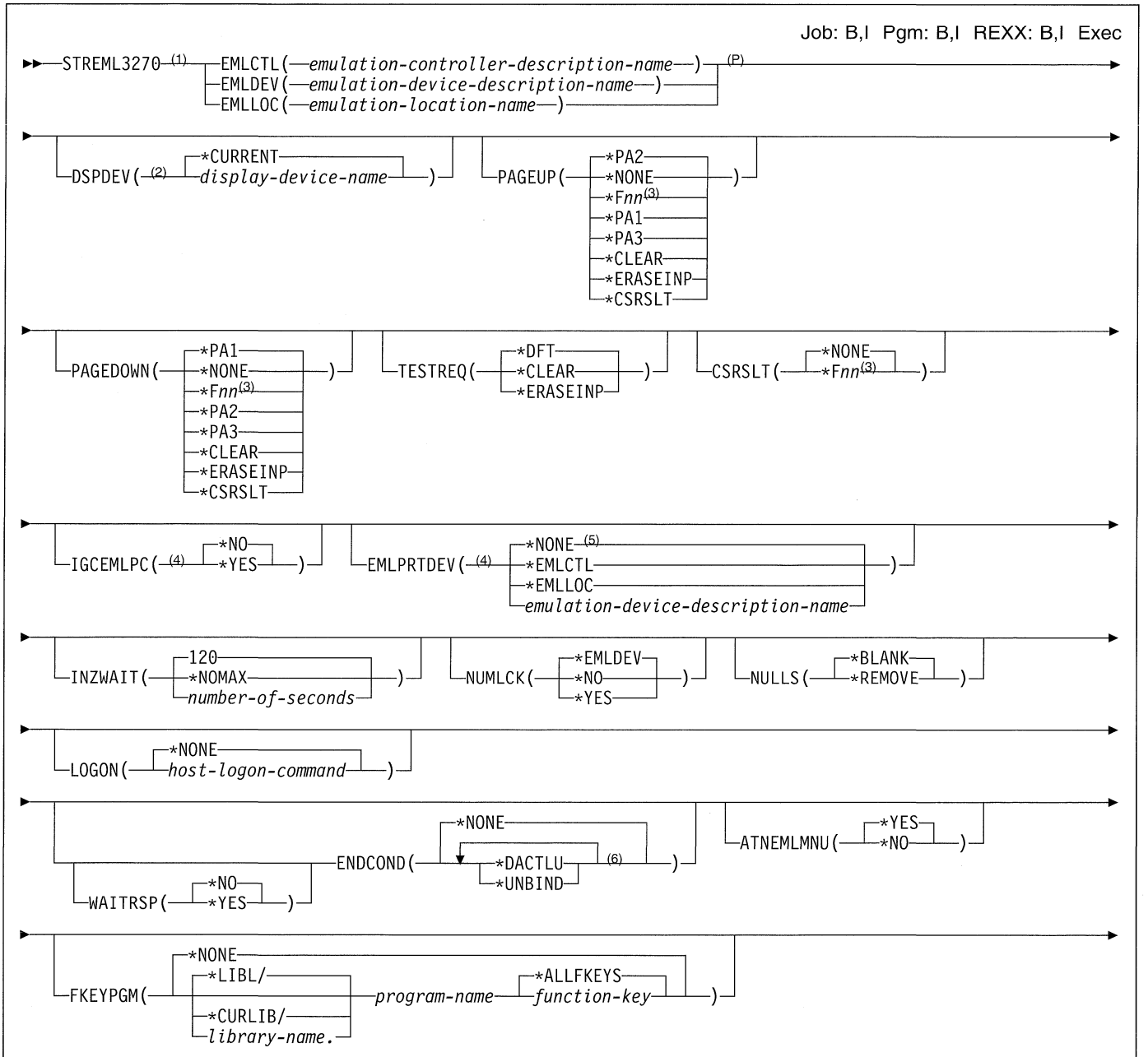
The `Start Education Administration` menu is shown for the Administrator.

The `Select Course Option` menu is shown for the new student that was enrolled by the Administrator.

The `Specify your Name` data entry screen is shown for the new student that was not enrolled.

The `Select Course Option` menu is shown for the enrolled student.

STREML3270 (Start 3270 Display Emulation) Command



Notes:

1 One parameter from the choices EMLCTL, EMLDEV, or EMLLOC is required.

2 DSPDEV is required when the command is a batch job. DSPDEV is ignored when the command is an interactive job.

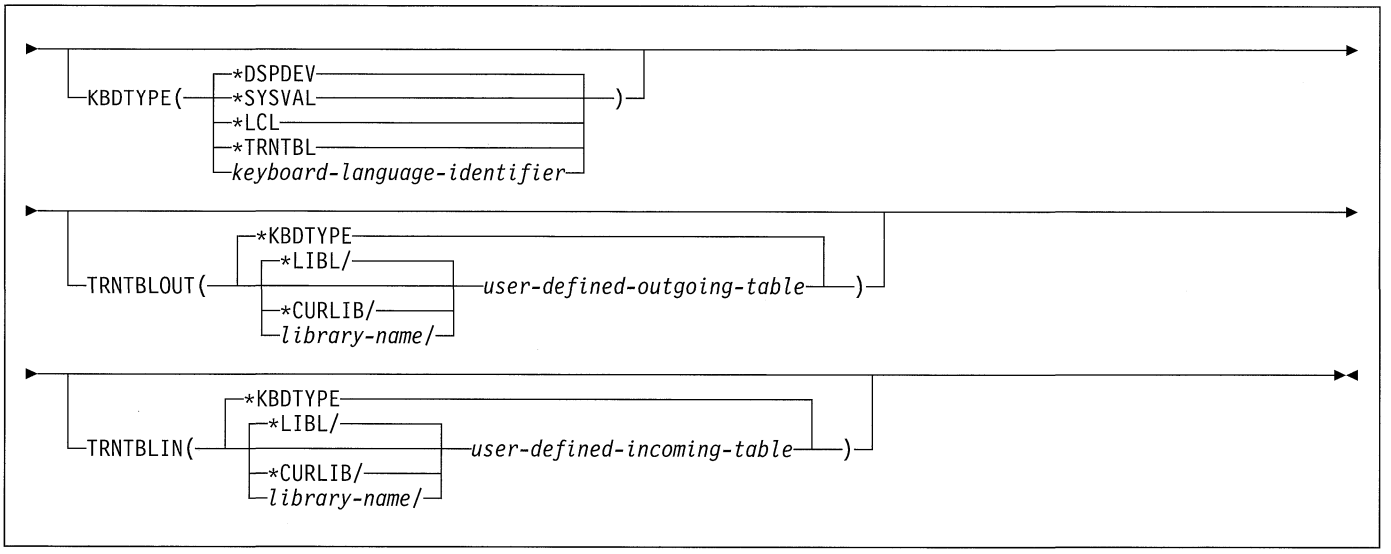
3 Where *nn* ranges from 1 through 24.

4 DBCS systems only.

5 EMLPRTDEV(*NONE) is valid only if IGCEMLPC(*NO) is specified.

6 Select one or more, a maximum of two repetitions.

P All parameters preceding this point can be specified in positional form.



Purpose

The Start 3270 Display Emulation (STREML3270) command starts a 3270 display emulation session. The user can type this command on the command line or from any display station that allows CL commands to be specified. This command can also be specified from a batch job by using the DSPDEV parameter on this command.

The STREML3270 command can be in a CL program specified as the INLPGM for a user profile that is run when the user signs on the display device.

More information on device emulation is in the *3270 Device Emulation Guide*.

Required Parameters

EMLCTL

Specifies the name of a binary synchronous communications (BSC) controller description (APPTYPE(*EML)) or Systems Network Architecture (SNA) controller description type of HOST with attached 3270 emulation display devices. When this parameter is specified, the requesting display device is linked to an available 3270 emulation display device on the emulation controller. The controller of the device must be available, and the requester of the command must be authorized to use the controller.

EMLDEV

Specifies the name of a BSC or an SNA emulation display device that is linked to the requesting display device to emulate a specific type of 3270 display device. The requester must be authorized for this device and the device must be available.

EMLLOC

Specifies the emulation remote location name that describes the location of the 3270 emulation display devices. When this parameter is specified, the requesting display device is linked to an available 3270 display device referred to by the remote location. At least one of the emulation devices referred to by the remote location must be available, and the requester of the command must be authorized to use the device. A remote location can refer to as many as 1,016 emulation display devices.

Optional Parameters

DSPDEV

Specifies the name of the display device used for display emulation when the command is in a batch job. The 3270 support tries to acquire the display device by this name; if the display device is acquired, the 3270 display emulation is active on that display device.

***CURRENT:** The current display device name is used for display emulation.

display-device-name: Specify the display device name used for display emulation.

PAGEUP

Specifies an added function for the Page Up (Roll Down) key on the 5250 keyboard when 3270 display emulation is active. This assignment is in effect when the number is not larger than the maximum number of input fields.

***PA2:** The 3270 PA2 function is assigned to the Page Up key.

***NONE:** No 3270 function is assigned to the Page Up key. When there are fewer input fields on the display than allowed by the 5250 display device, the Page Up key has no function.

***F-KEY:** Specify the 3270 function key assigned to the Page Up key.

***PA1-KEY:** Specify the PA1 key assigned to the Page Up key.

***PA3-KEY:** Specify the PA3 key assigned to be the Page Up key.

***CLEAR:** Specify the 3270 Clear function assigned to the Page Up key.

***ERASEINP:** Specify the 3270 Erase Input function key assigned to the Page Up key.

***CSRSLT:** Specify the 3270 Cursor Select Key function key assigned to the Page Up key, and the real Cursor Select Key cannot be used.

PAGEDOWN

Specifies an added function for the Page Down (Roll Up) key on the 5250 keyboard when 3270 display emulation is active. This assignment is in effect when the number is not larger than the maximum number of input fields.

***PA1:** The 3270 PA1 function is assigned to the Page Down key.

***NONE:** No 3270 function is assigned to the Page Down key. When there are fewer input fields on the display than allowed by the 5250 display device, the Page Down key has no function.

***F-KEY:** Specify the 3270 function assigned to the Page Down key.

***PA2-KEY:** Specify the PA2 key assigned to the Page Down key.

***PA3-KEY:** Specify the PA3 key assigned to be the Page Down key.

***CLEAR:** Specify the 3270 Clear function assigned to the Page Down key.

***ERASEINP:** Specify the 3270 Erase Input function assigned to the Page Down key.

***CSRSLT:** Specify the 3270 Cursor Select Key function assigned to the Page Down key, and the real Cursor Select Key cannot be used.

TESTREQ

Specifies an added function for the Test Request key on the 5250 keyboard when a 3270 display emulation is active.

***DFT:** Normal function is assigned to the Test Request key. This is the system default. The normal function is defined in the emulation Help text, and depends on whether the 3270 emulation display device is BSC or SNA. BSC defaults to a 3270 test request function, while SNA defaults to a 3270 system request function.

***CLEAR:** The Clear key is assigned to the Test Request key.

***ERASEINP:** The Erase Input key is assigned to the Test Request key.

CSRSLT

Specifies a function key to emulate the cursor select key.

***NONE:** A physical function key is not assigned to emulate the cursor select key. Instead, the Cursor Select Key defined on the keyboard is used.

***F-KEY:** Specify the function key assigned to emulate the Cursor Select Key, and the Cursor Select Key defined on the keyboard cannot be used.

IGCEMLPC

Specifies whether SNA Japanese 3270 PC emulation or 3270 device emulation is used.

***NO:** SNA Japanese 3270 PC emulation is not used.

***YES:** SNA Japanese 3270 PC emulation is used.

EMLPRTDEV

Specifies the printer device used for SNA Japanese 3270 PC emulation. The printer device is selected after the display device is selected.

***NONE:** SNA Japanese 3270 PC emulation with display emulation is not used. No emulation printer device is selected.

***EMLCTL:** The first available emulation printer device from the specified emulation controller in the EMLCTL parameter is used.

***EMLLOC:** The first available emulation printer device from the specified emulation location in the EMLLOC parameter is used.

emulation-device-description-name: Specify the emulation printer device with the selected emulation display device for SNA Japanese 3270 PC emulation.

INZWAIT

Specifies the first amount of time (in seconds) that the utility waits for the first display from the host system. This parameter only applies to the first read to the host system. If the host system does not send the first display within this time, the emulation session is ended and a message is returned to the requester.

120: The utility waits 120 seconds for the first display from the host system.

***NOMAX:** There is no limit on the amount of time the utility waits for the first display from the host system. This value can be used when the user is not sure when the host system is active to this session. The request can be ended by using the system request and ending request functions.

number-of-seconds: Specify the length of time (in seconds) that the utility waits for the first display from the host system. Valid values range from 1 through 32767.

NUMLCK

Specifies whether the numeric input fields allow only numeric data on a 5250 keyboard.

***EMLDEV:** Numeric shift lock is specified in the emulation device description in the EMLNUMLCK parameter. The DSPDEVD command can be used to display the current EMLNUMLCK setting for the emulation device.

***NO:** Data can be typed in the numeric input fields of a 3270 emulation device.

***YES:** Only numeric data can be typed in the numeric input fields of a 3270 emulation device. Numeric data includes characters ranging from 0 through 9, and symbols + - , . and b.

NULLS

Specifies how beginning and embedded nulls within the 3270 data stream (sent from a 5250 display station) are changed.

***BLANK:** Beginning and embedded nulls are changed to blanks before the 3270 data stream is sent to the host system.

***REMOVE:** Beginning and embedded nulls are removed before the 3270 data stream is sent to the host system.

LOGON

Specifies the sign-on (logon) text that is sent to the host system after starting SNA 3270 display emulation. This text can be used to sign on a specific host application.

This parameter is not allowed if specified for BSC 3270 display emulation, SNA 3270 display station pass-through, or SNA DBCS 3270PC emulation.

***NONE:** No text is sent to the host system after starting 3270 display emulation.

host-logon-command: Specify text that is sent to the host system after starting 3270 display emulation. The text must be enclosed in apostrophes if it contains blanks or other special characters. All apostrophes within the text must be represented by two apostrophes. A maximum of 256 characters can be specified.

WAITRSP

Specifies whether the 3270 emulation device waits until the data received is shown on the workstation display to send a positive response to the host system. The response time recorded by the AS/400 system may be longer than the time recorded by the host when the emulation device does not wait.

***NO:** The emulation device does not wait to send a positive response. It sends the response as soon as the data received is sent to the workstation display.

***YES:** The emulation device waits until the data received is shown on the workstation display to send a positive response.

ENDCOND

Specifies additional ways in which the SNA 3270 display emulation session can end.

This parameter is not allowed if BSC 3270 display emulation, SNA 3270 display station pass-through, or SNA DBCS 3270PC emulation is specified on the EMLDEV parameter.

***NONE:** No additional ways to end 3270 display emulation are requested.

***DACTLU:** The 3270 display emulation session ends if it receives an SNA DACTLU command from the host system.

There are certain host system applications that issue a DACTLU (deactivate logical unit) before starting, such as Time Sharing Option (TSO), which ends the 3270 display emulation session before the desired application is accessed. This end condition must be avoided when trying to access these applications.

***UNBIND:** The 3270 emulation session ends if it receives an SNA UNBIND from the host system. Please consider the following items before selecting this end condition:

- This end condition can be used only when accessing one host application for the duration of the session. An UNBIND occurs while switching from one application to the next, and the 3270 session ends before accessing the second application.
- This end condition can only be used when the communication path to the host system is a simple one. A simple communication path is one that only involves accessing the AS/400 system where the Start 3270 Display Emulation (STREML3270) command is run, and accessing the host system that contains the desired application. Intermediate systems can exist along this simple path as long as they are not accessed. If intermediate systems are accessed, an UNBIND occurs while switching from one system to the next, and the 3270 display emulation session ends before accessing the desired application.
- There are certain host system applications that issue an UNBIND before starting, such as Time Sharing Option (TSO), which ends the 3270 display emulation session before the desired application is accessed. This end condition must be avoided when trying to access these applications.

ATNEMLMNU

Specifies whether or not the emulation menu is displayed when the Attn key is pressed.

***YES:** The 3270 display emulation menu is displayed when the Attn key is pressed.

***NO:** The 3270 display emulation menu is not displayed when the Attn key is pressed.

FKEYPGM

Specifies a user-exit program and one or more function keys that call the program. When a specified function key is pressed during the 3270 display emulation session and is sent to the host system, the user-exit program is called. When the user-exit program ends, control is returned to the 3270 display emulation session at the point where the function key was pressed.

This parameter is not valid if specified for either BSC 3270 display emulation, SNA 3270 display station pass-through, or SNA DBCS 3270PC emulation.

Note: The user-exit program is called only if the function key is successfully sent to the host system.

If the function key fails to be received, an error reset message appears at the bottom of the display suggesting you to try again.

The AS/400 user-exit program must be coded to allow for input parameters. The following parameters are passed to the program in the specified order:

1. The function key identifier (10 characters). The identifier of the function key that was pressed. If function key 1 is pressed, the parameter value is *F1. If function key 2 is pressed, the parameter value is *F2, and so on, up to function key 24. The value is left-justified within the parameter.
2. The display name (10 characters). The name of the display on which the 3270 display emulation session is running. The value is left-justified within the parameter.
3. The cursor location (6 characters). The screen location of the cursor at the time the function key was pressed. The first three characters are the row position of the cursor location. The second three characters are the column position of the cursor location. For example, if the cursor location is row 24, column 1 when the function key is pressed, the value of the parameter is 024001. The row and column can be extracted from the variable using substring logic.

Element 1: Program Name

***NONE:** A user-exit program is not associated with any function key.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name and library of the user-exit program that is called when one of the specified function keys is pressed. The program cannot be a system program.

Element 2: Function Key

***ALLFKEYS:** All function keys call the specified user-exit program.

function-key: Specify a function key to call the user-exit program. A maximum of 24 values can be specified on this parameter.

KBDTYPE

Specifies the type of keyboard language used on the display device.

This parameter does not apply when running SNA 3270 display station pass-through.

***DSPDEV:** The display device description is used.

***SYSVAL:** The value specified in the system value QKBDTYPE is used.

***LCL:** The display device that requested 3270 display emulation is a local display device. The keyboard type is determined from the display device description.

***TRNTBL:** Allows user-defined translate tables to be used. The character translation is defined in the translation tables specified by the TRNTBLOUT and TRNTBLIN parameters. This value is valid for both local and remote display devices.

keyboard-language-identifier: Specify the 3-character identifier for the language group associated with the remote display requesting 3270 emulation. The identifier consists of 3 characters. The languages and associated identifiers are shown below.

Table 74 (Page 1 of 2). Keyboard Mapping Table

Language/Country	Identifier	ASCII Device Group
Arabic X/Basic	CLB	D
Austria/Germany	AGB	A, B
Austria/Germany Multinational	AGI	A, B
Belgium Multinational	BLI	B
Brazilian Portuguese	BRB	
Canadian French	CAB	A, B
Canadian French Multinational	CAI	A, B
Cyrillic	CYB	
Denmark	DMB	B
Denmark Multinational	DMI	B
Finland/Sweden	FNB	B
Finland/Sweden Multinational	FNI	B
France (Azerty)	FAB	A, B
France (Azerty) Multinational	FAI	A, B
France (Qwerty)	FQB	
France (Qwerty) Multinational	FQI	
Greece	GNB	
Greece	GKB	
Hebrew	NCB	D
Iceland	ICB	
Iceland Multinational	ICI	
International	INB	
International Multinational	INI	
Italy	ITB	A, B
Italy Multinational	ITI	A, B
Japan English	JEB	
Japan English Multinational	JEI	
Japan Kanji (For PS*/55 and 5295 display stations)	JKB	
Japan United States Basic	JUB	
Japan Katakana (For 5251, 5291, 5292, and 3180 Katakana display stations)	KAB	
Korea	KOB	

Table 74 (Page 2 of 2). Keyboard Mapping Table

Language/Country	Identifier	ASCII Device Group
Latin-2/ROECE	ROB	
Netherlands	NEB	
Netherlands Multinational	NEI	
Norway	NWB	B
Norway Multinational	NWI	B
Portugal	PRB	B
Portugal Multinational	PRI	B
Simplified Chinese	RCB	
Spain	SPB	B
Spain Multinational	SPI	B
Spanish Speaking	SSB	B
Spanish Speaking Multinational	SSI	B
Sweden	SWB	B
Sweden Multinational	SWI	B
Switzerland/French Multinational	SFI	B
Switzerland/German Multinational	SGI	B
Thai	THB	
Traditional Chinese	TAB	
Turkey	TKB	
United Kingdom	UKB	A, B
United Kingdom Multinational	UKI	A, B
United States/Canada	USB	A, B, C
United States/Canada Multinational	USI	A, B, C
Languages of the former Yugoslavia	YGI	

Note: For example, KBDTYPE(USB) indicates a keyboard using the basic United States/Canada character set.

TRNTBLOUT

Specifies the outgoing translation table used to translate characters sent from the host system to 3270 emulation. This parameter is valid only when KBDTYPE(*TRNTBL) is specified. Both TRNTBLOUT and TRNTBLIN must be specified if KBDTYPE(*TRNTBL) is specified.

***KBDTYPE:** Translation is done using the language specified in the KBDTYPE parameter.

The name of the outgoing translation table can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

user-defined-outgoing-table: Specify the qualified name of the table used for outgoing translation.

TRNTBLIN

Specifies the incoming translation table used to translate characters sent from 3270 emulation to the host system. This parameter is valid only when KBDTYPE(*TRNTBL) is specified. Both TRNTBLOUT and TRNTBLIN must be specified if KBDTYPE(*TRNTBL) is specified.

***KBDTYPE:** Translation is done using the language specified in the KBDTYPE parameter.

The name of the incoming translation table can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

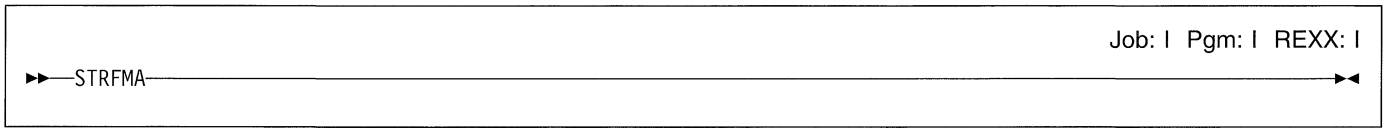
user-defined-incoming-table: Specify the qualified name of the table used for incoming translation.

Example

```
STREML3270   EMLCTL(TSOHOST)  PAGEUP(*F7)  PAGEDOWN(*F8)
```

This command places the display device into an emulation session that uses the first available device on the controller description TSOHOST for which the user has authority. When there are fewer input fields on the display than the maximum allowed by the 5250 display device and the Page Up key is pressed, an F7 key value is sent to the host system. When the Page Down key is pressed, an F8 key value is sent to the host system.

STRFMA (Start Font Management Aid) Command



Purpose

The Start Font Management Aid (STRFMA) command shows the main Font Management Aid (FMA) menu. From this menu you can select the following options:

- Work with user-defined characters from the work station font file to the double-byte character set (DBCS) font table.
- Copy user-defined characters from the DBCS font table to the work station font file.
- Distribute the dictionary or the font file of the work station.

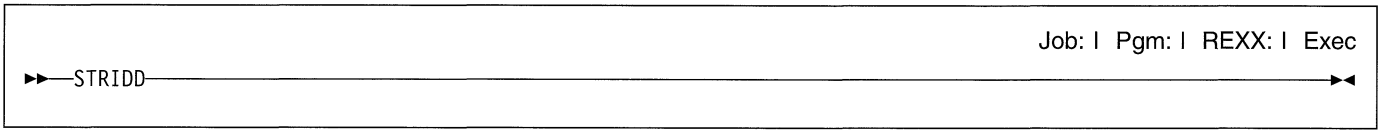
There are no parameters for this command.

Example

```
STRFMA
```

This command displays the main FMA menu.

STRIDD (Start Interactive Data Definition Utility) Command



Purpose

The Start Interactive Data Definition Utility (STRIDD) command displays the main IDDU menu. From this menu, you can select an option that allows you to work with data definitions, data dictionaries, files, and libraries, or use related commands and office tasks.

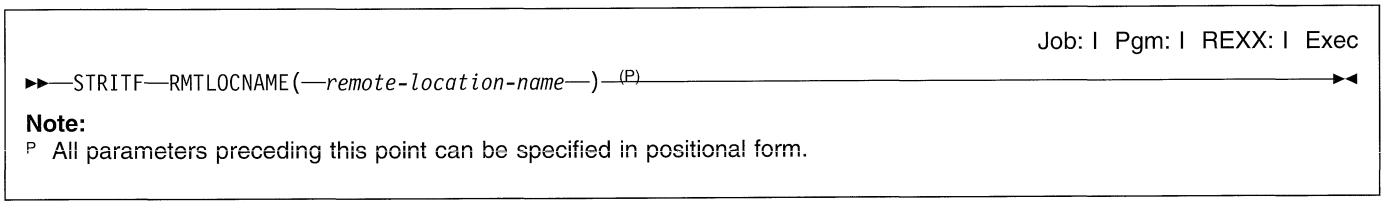
There are no parameters for this command.

Example

```
STRIDD
```

This commands displays the main IDDU menu.

STRITF (Start Interactive Terminal Facility) Command



Purpose

The Start Interactive Terminal Facility (STRITF) command allows the user to send and receive data and file members for asynchronous display stations. The system user can also send documents. The user must vary on asynchronous communications descriptions before the Interactive Terminal Facility (ITF) can be used.

Required Parameters

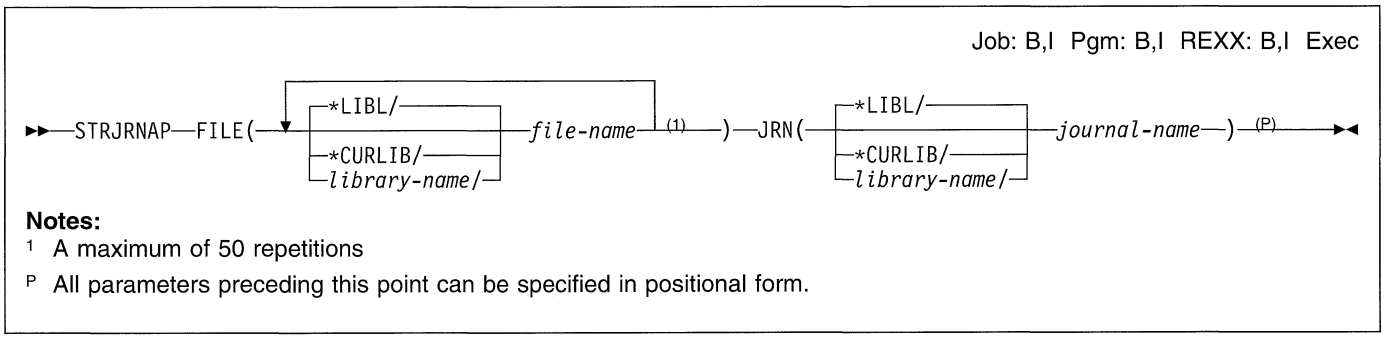
RMTLOCNAME
Specifies the remote location name of the system with which this object communicates.

Example

```
STRITF  CHICAGO
```

This command allows the user to communicate with the remote location CHICAGO.

STRJRNAP (Start Journal Access Path) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Start Journal Access Path (STRJRNAP) command is used to start the journaling of the access path, or access paths, for all members of a database file to a specific journal. The access paths of new members are also journaled.

If a physical file is specified, journaling can be started for its access paths. When access path journaling is started for a physical file, only the access paths for the physical file members are journaled. Journaling for any logical file access paths is started only when access path journaling is started for the logical file.

The journaled entries created after running this command cannot be used in any apply or remove journal changes operation. These entries are used only to recover the access path without rebuilding it after an abnormal system operation ending.

Restrictions:

1. Before journaling an access path, all physical files over which the access path is built must first be journaled to the same journal that is used to journal the access path. Even if all physical file members for a particular physical file are removed from the access path of a logical file, all physical files must still be journaled to the same journal before journaling the access path.
2. All access paths to be journaled must specify MAINT(*IMMED) or MAINT(*DLY) and FRCACCPTH(*NO).
3. If only *after* images are being journaled for the physical file members, the system automatically starts journaling the before and after images for the physical file once journaling is started for any access path built over the physical file. When journaling ends for the access paths, the system automatically stops journaling the before images for the physical file and again only journals the after images.
4. Overrides are not applied to files specified on the FILE parameter.

Required Parameters

FILE

Specifies a maximum of 50 qualified names of the database files for which access paths are to be journaled.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the file for which access paths are to be journaled.

JRN

Specifies the qualified name of the journal that is to receive the journaled access path changes.

The name of the journal can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

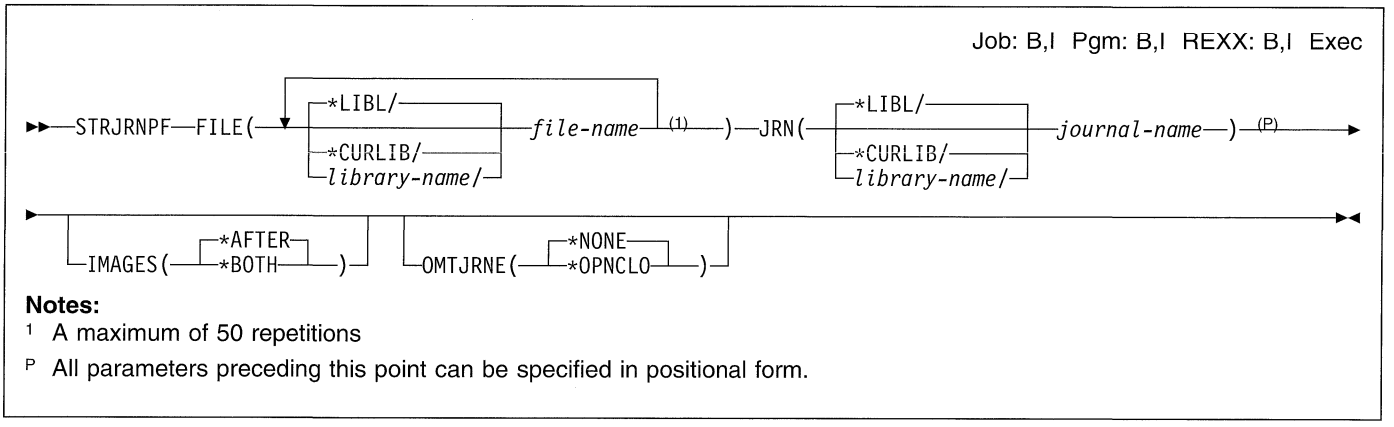
journal-name: Specify the name of the journal that is to receive the journaled access path changes.

Example

```
STRJRNAP  FILE(MYFILE)  JRN(MYLIB/JRNLA)
```

This command journals all access paths for all members in file MYFILE (found using the library search list) to journal JRNLA in library MYLIB.

STRJRNPF (Start Journal Physical File) Command



Notes:
1 A maximum of 50 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Start Journal Physical File (STRJRNPF) command is used to start journaling changes (made to all members of a specific physical file) to a specific journal. Changes in new members added to the file are also journaled to that journal.

The user can specify that only the *after* image or both *before* and *after* images of records in the journaled physical file be journaled. *Before* images are necessary for operation of the IBM-supplied rollback command, Remove Journaled Changes (RMVJRNCHG).

After journaling begins for the file, and after any new members are added to the file, the user should run the Save Changed Object (SAVCHGOBJ) command with OBJTYPE(*FILE) and OBJJRN(*YES) specified. The file must be saved because journaled changes cannot be applied to a version of the file that was saved before journaling was in effect.

Restrictions:

1. The file must not be journaling changes to another journal.
2. No members of the file currently on the system can be in use.
3. Overrides are not applied to files specified on the FILE parameter.
4. The maximum number of objects that can be associated with one journal is 262,136. This maximum includes physical file members whose changes are currently being journaled, members for which journaling was ended while the current receiver was attached, and journal receivers that are or were associated with the journal while the current journal receiver is attached. If the number of objects is larger than this maximum, journaling does not start.

Required Parameters

FILE

Specifies a maximum of 50 qualified names of the physical files for which changes are to be journaled.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the file for which changes are to be journaled.

JRN

Specifies the qualified name of the journal that receives the journaled changes.

The name of the journal can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

journal-name: Specify the name of the journal that receives the journaled changes.

Optional Parameters

IMAGES

Specifies the kinds of record images that are written to the journal receiver for updates to records in the file.

***AFTER:** Only *after* images are generated for changes to records in this file.

***BOTH:** The system generates both *before* and *after* images for updates to records in this file.

OMTJRNE
Specifies the journal entries that are omitted.

***NONE:** No entries are omitted.

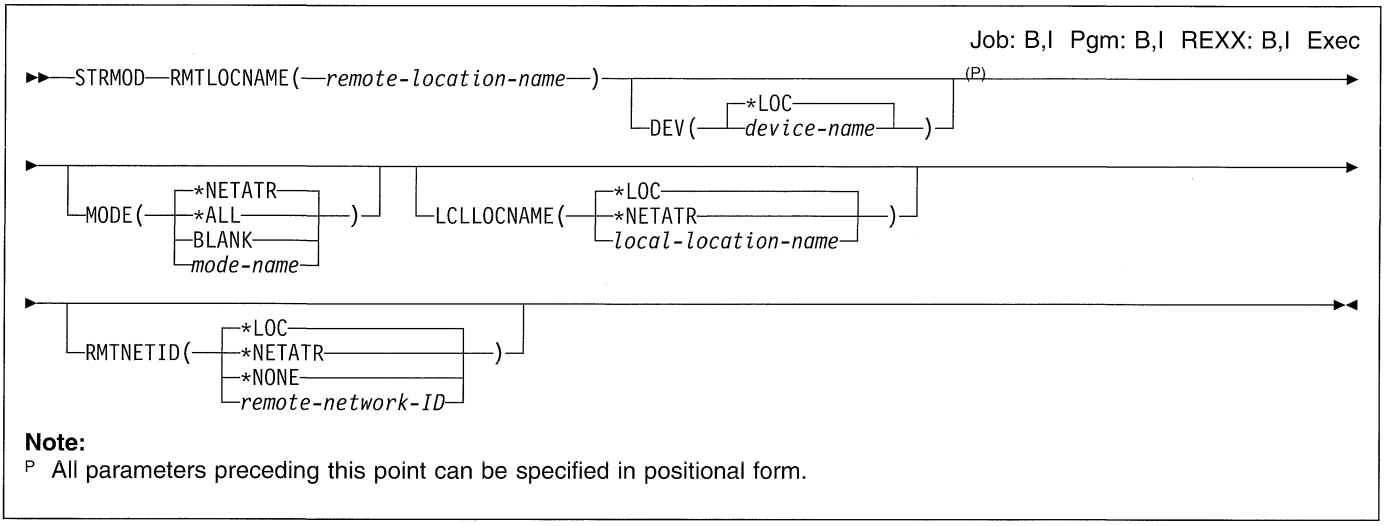
***OPNCLO:** Open and close entries are omitted. Open and close operations on the specified file members do not generate open and close journal entries. This prevents the use of TOJOBO and TOJOBC entries on the Apply Journal Changes (APYJRNCHG) and Remove Journal Changes (RMVJRNCHG) commands, but it saves some storage space in the attached journal receivers.

Example

```
STRJRNPF  FILE(MYFILE)  JRN(MYLIB/JRNLA)
```

This command journals all changes to all members of file MYFILE (as found using the library search list) to journal JRNLA in library MYLIB. Only the *after* images of updated records are written to the journal.

STRMOD (Start Mode) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Start Mode (STRMOD) command starts one or all modes currently in use for an advanced program-to-program communications (APPC) remote location. The user can use STRMOD in either the reset or ended state; it is required only after an End Mode (ENDMOD) command has ended a mode. More information is in the *APPC Programmer's Guide*.

Restriction: The user must have operational authority for the APPC device to use this command.

Required Parameters

RMTLOCNAME

Specifies the remote location name for which one or more modes is being started. Specify the name of the remote location.

Optional Parameters

DEV

Specifies the device description used with the remote location.

***LOC:** The device associated with the remote location is used. If several devices are associated with the remote location, the system determines which device is used.

device-name: Specify the name of the device being used.

MODE

Specifies the mode that is started.

***NETATR:** The mode name specified in the network attributes is used.

***ALL:** Specifies all modes currently in use by the remote location are being started.

BLANK: The mode name consisting of 8 blank characters is used.

mode-name: Specify the mode name being started.

LCLLOCNAME

Specifies the local location name.

***LOC:** The device associated with the remote location is used. If several devices are associated with the remote location, the system determines which device is used.

***NETATR:** The LCLLOCNAME value specified in the system network attributes is used.

local-location-name: Specify the name of the local location. This name is specified if the user wants to indicate a specific local location name to be associated with the remote location.

RMTNETID

Specifies the remote network ID used with the remote location.

***LOC:** The remote network identifier (ID) associated with the remote location is used. If several remote network IDs are associated with the remote location, the system determines which remote network ID is used.

***NETATR:** The LCLNETID value specified in the system network attributes is used.

***NONE:** No remote network identifier (ID) is used.

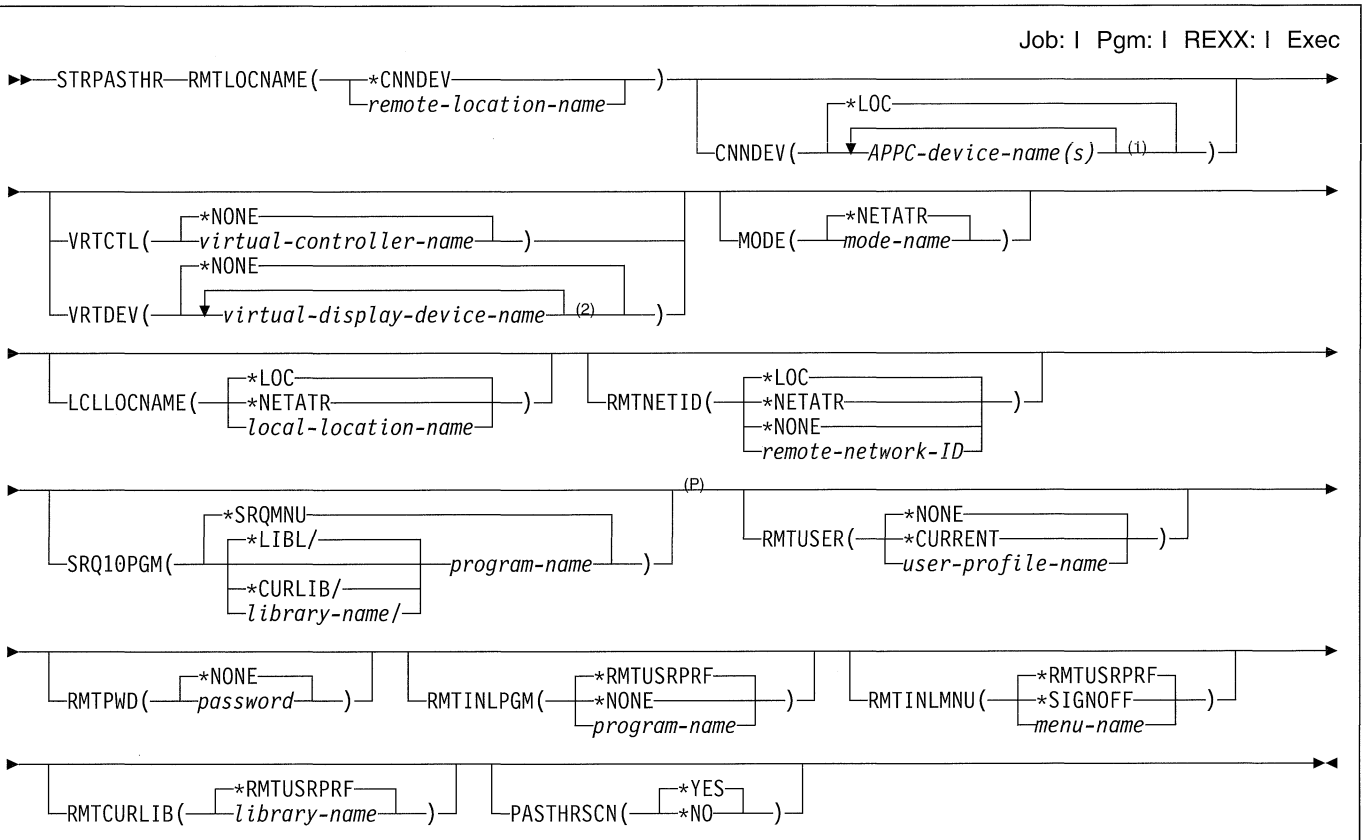
remote-network-ID: Specify the name of the remote network ID used.

Example

```
STRMOD  RMTLOCNAME(APPCRLOC)  DEV(APPCDEV)
  MODE(APPCMODE)  RMTNETID(CHICAGO)
```

This command starts a mode named APPCMODE for a remote location named APPCRLOC, a device named APPCDEV, and a remote network ID of CHICAGO.

STRPASTHR (Start Pass-Through) Command



Notes:
1 A maximum of 16 repetitions
2 A maximum of 32 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Start Pass-Through (STRPASTHR) command allows pass-through (and sign-on) to a remote system as if attachment is local. This command uses the AS/400 system advanced program-to-program communications (APPC) and advanced peer-to-peer networking (APPN) support to connect the local display station to the remote system. More information on configuring or operating the pass-through operation is in the *Remote Work Station Guide*.

Restrictions: (1) This command is not valid if it is entered in a job that is already in pass-through mode on a remote system. (2) This command cannot be entered at a work station with a display that has 12 lines by 80 characters.

Required Parameters

RMTLOCNAME
Specifies, if CNNDEV(*LOC) is specified, the name of the remote location that is the target of the pass-through session, or if devices are specified on the CNNDEV parameter, it specifies the first system that does intermediate pass-through routing.

***CNNDEV:** The APPC devices specified on the CNNDEV parameter are used.

remote-location-name: Specify the name of the remote location that is the target of the pass-through session or the name of the first system that does intermediate pass-through routing. The advanced peer-to-peer networking (APPN) support function determines the route to this location.

Optional Parameters

CNNDEV
Specifies the names of the device descriptions that connect the first system that does pass-through routing from the source system to the target system. If RMTLOCNAME(*CNNDEV) is specified, the first device specified on this parameter is on the source system. If the RMTLOCNAME parameter is a location, the first device specified is on the system containing that location. If the target system is more than one system

away and pass-through must establish the intermediate sessions, a list of APPC device descriptions must be specified. The APPC device names must be listed in the order that the systems are passed through to reach the target system.

***LOC:** The RMTLOCNAME parameter is used to identify the target of the pass-through session and the path used to reach the target.

APPC-device-name(s): Specify the names of the device descriptions that complete the route from the source system to the target system. Up to 16 names can be specified.

VRTCTL

Specifies the name of the virtual controller on the remote system that is used to do a pass-through session. If a virtual controller is specified, one of the virtual display devices attached to it is selected to do the pass-through session. The system tries to compare the device type and model of the physical display device with an available virtual device. If the same type or a similar type is not available, a comparison is made with a 5251 Model 11, if available. Graphics cannot be done with a 5251 Model 11 device; a 5292 Model 2 device is required. If no virtual devices are available, another virtual controller must be specified, or the user must try again later.

***NONE:** No controller is specified. VRTDEV(*NONE) requests that the target system automatically configures a virtual device.

virtual-controller-name: Specify the name of the virtual controller description on the remote system.

VRTDEV

Specifies one or more devices on the remote system that are connected to a virtual controller used for the pass-through session. A device from the list on the remote system is selected based on a comparison of device type and model. If more than one device is in the list, the first available device that closely matches the type and model of the device on which the command is entered is used. If the same type or a similar type is not available, a comparison is made with a 5251 Model 11, if available. Graphics cannot be done with a 5251 Model 11 device; a 5292 Model 2 device is required.

***NONE:** No device names are specified. VRTCTL(*NONE) requests that the target system automatically configures a virtual device.

virtual-display-device-name: Specify the names of the virtual display device descriptions on the remote system used for the pass-through session. Up to 32 names can be specified.

MODE

Specifies the mode name used for the pass-through session. This parameter is not valid if RMTLOCNAME(*CNNDEV) is specified. In this case, the system uses the first mode name specified in the APPC device description.

***NETATR:** The pass-through mode in the network attributes is used for the pass-through session.

mode-name: Specify a mode name to use for the pass-through session. Specify BLANK for a mode name consisting of eight blank characters.

LCLLOCNAME

Specifies the local location name.

***LOC:** The device associated with the remote location is used. If several devices are associated with the remote location, the system determines which device is used.

***NETATR:** The default location name defined in the Change Network Attributes (CHGNETA) command is used.

local-location-name: Specify the local location name for the source system.

RMTNETID

Specifies the network identifier (ID) of the network in which the remote location resides. This parameter is not valid if RMTLOCNAME(*CNNDEV) is specified.

***LOC:** The remote network identifier (ID) associated with the remote location is used. If several remote network IDs are associated with the remote location, the system determines which remote network ID is used.

***NETATR:** The RMTNETID value specified in the system network attributes is used.

***NONE:** No remote network identifier (ID) is used.

remote-network-ID: Specify the identifier of the network in which the target system (specified by the RMTLOCNAME parameter) is located.

SRQ10PGM

Specifies that the System Request menu or a user-written program starts when SYSREQ option 10 is selected. The user program displays a menu that allows selection of the system to access, and then transfers to a group job that sends the STRPASTHR command to the desired system. More information is in the *Remote Work Station Guide*.

***SRQMNU:** The System Request menu is displayed.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of the program that starts when the SYSREQ option 10 is selected.

RMTUSER

Specifies the user profile for automatic sign-on to the target system. If a profile is specified for this parameter and password security is active on the target system, RMTPWD(*NONE) is not valid.

***NONE:** No user profile name is sent, and automatic sign-on does not occur.

***CURRENT:** The user profile under which the current job is running is used.

user-profile-name: Specify a user profile name to use that exists on the target system. If the target system allows it, and the user profile exists on the target system, the user is automatically signed on. Otherwise, the user is presented with a sign-on display on the target system or a failure message on the source system, depending upon the configuration of the target system. If a profile is specified and password security is active on the target system, a password must be specified, even if the profile specified is the same as the current profile.

RMTPWD

Specifies the password sent to the target system.

***NONE:** The system does not pass a password. If a profile is specified on the RMTUSER parameter and password security is active on the target system, this value is not allowed.

password: Specify a password sent to the target system to verify the sign-on of the user specified in the RMTUSER parameter. This password sent is not encrypted across the communication line.

RMTINLPGM

Specifies the program called immediately after sign-on to the system.

***RMTUSRPRF:** The initial program specified in the remote user profile is run immediately after the job (which was automatically signed-on) starts.

***NONE:** No program is run before the first menu is displayed, even if the first program is specified in the remote user profile.

program-name: Specify the name of a program that is run immediately after automatic sign-on.

RMTINLMNU

Specifies the first menu shown when automatically signed on the target system after the first program is run.

***RMTUSRPRF:** The initial menu specified in the remote user profile is shown immediately after the first program is run.

***SIGNOFF:** No menu is displayed after the first program is run, even if an initial menu is specified in the remote user profile. After the program runs, the user is signed off, and the program runs.

menu-name: Specify the menu shown immediately after the initial program is run.

RMTCURLIB

Specifies the name of the library that becomes the current library in the library list of the job after automatic sign-on to the system.

***RMTUSRPRF:** The current library specified in the remote user profile becomes the current library in the library list after automatic sign-on.

library-name: Specify the name of the library that becomes the current library in the library list before the pass-through session is established.

PASTHRSCN

Specifies whether the pass-through display and associated status messages appear before the pass-through session is established.

***YES:** The pass-through display and informal messages are shown before the pass-through session is established.

***NO:** The pass-through display and information messages are not shown before the pass-through session is established.

Examples

Example 1: Pass-Through to Toronto

```
STRPASTHR  RMTLOCNAME(*CNNDEV)  CNNDEV(DET CHI TOR)
  VRTCTL(VWSC)
```

This command specifies starting a pass-through to the Toronto system by going through Detroit and Chicago. More information is in the *Remote Work Station Guide*.

Example 2: Pass-Through to Detroit

```
STRPASTHR  RMTLOCNAME(DETROIT)  VRTCTL(VWSC)
```

This command specifies a pass-through to the Detroit system. APPN establishes the route to Detroit.

Example 3: Pass-Through to Toronto

```
STRPASTHR  RMTLOCNAME(DETROIT)  CNNDEV(CHI TOR)
  VRTCTL(VWSC)
```

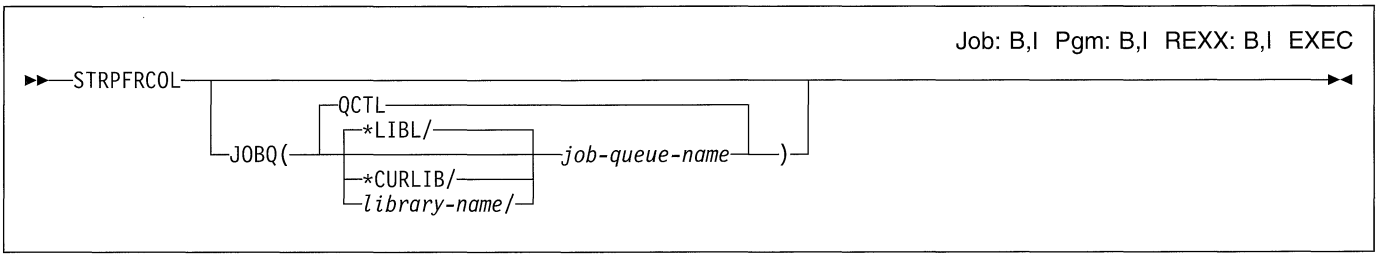
This command specifies another way to pass-through to the Toronto system by going through Chicago and Detroit. APPN establishes the route to Detroit.

Example 4: Pass-Through to Detroit

```
STRPASTHR  RMTLOCNAME(DETROIT)  RMTUSER(*CURRENT)
```

This command specifies a pass-through to the DETROIT system and an automatic sign-on using the user profile with the same name as the one currently used on the source system. It also specifies that the DETROIT system automatically configures a virtual device for the pass-through session, since a virtual controller or virtual device was not specified.

STRPFRCOL (Start Performance Collection) Command



Purpose

The Start Performance Collection (STRPFRCOL) command submits the QPFRCOL job that activates the automatic performance data collections schedule that is defined on the Work with Performance Collection (WRKPFRCOL) command.

Optional Parameter

JOBQ

Specifies the qualified name of the job queue on which this job is placed.

QCTL: The job is submitted to the QCTL job queue, which is the IBM-supplied job queue for the controlling subsystem.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

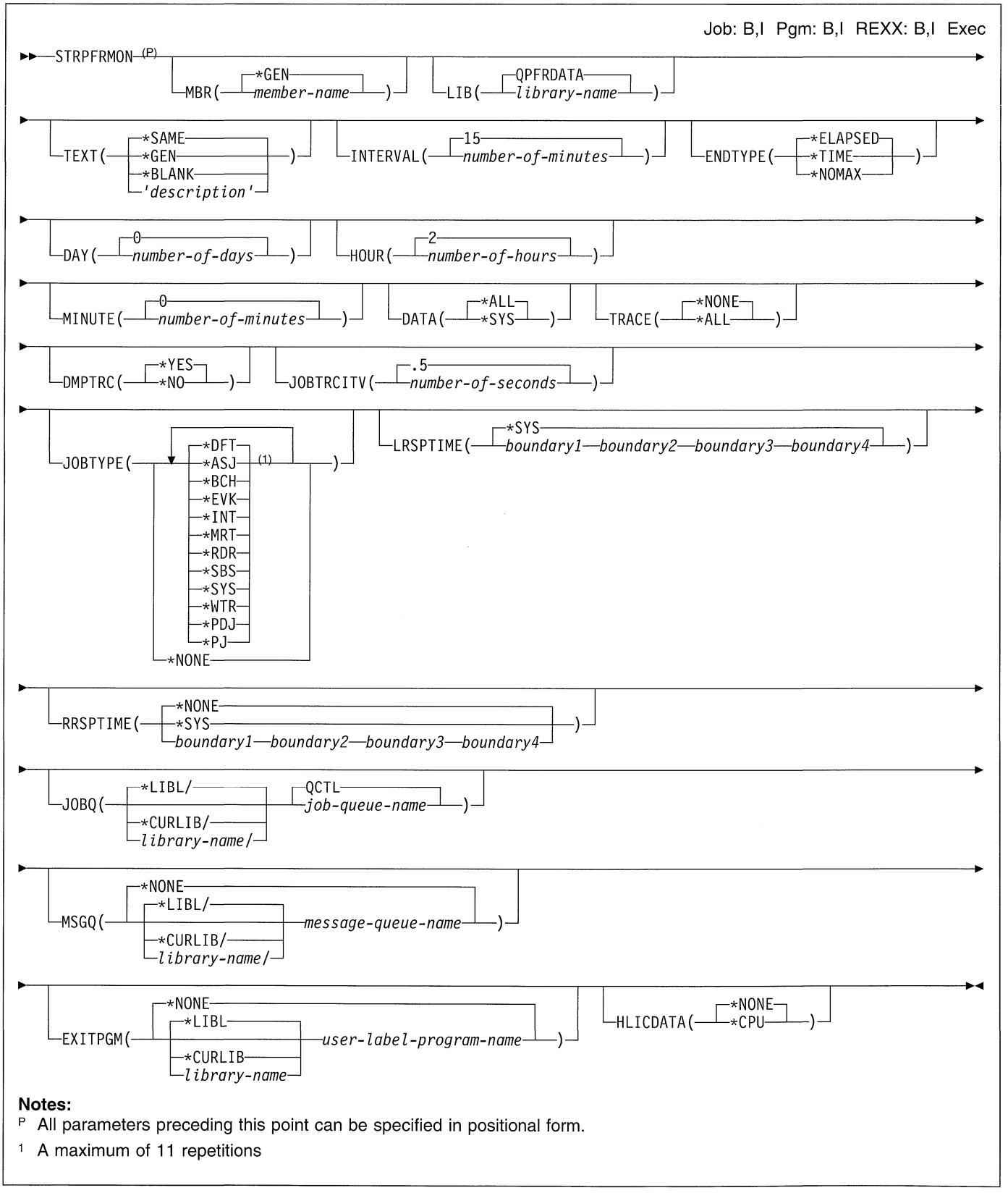
job-queue-name: Specify the qualified name of the job queue on which the submitted job is placed.

Example

```
STRPFRCOL JOBQ(QGPL/QBATCH)
```

This command submits the job starting the automatic performance collections schedule to the QBATCH job queue in the QGPL library.

STRPFRMON (Start Performance Monitor) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 A maximum of 11 repetitions

Purpose

The Start Performance Monitor (STRPFRMON) command submits the performance monitoring job (QPFRMON) to the job queue identified in the JOBQ parameter. Once the job is started, it periodically collects performance data and puts it into system-supplied database files. The data is collected at intervals specified in the INTERVAL parameter.

Optional Parameters

MBR

Specifies the database file member to which the output of the collected data is directed.

***GEN:** The member name is created by the system. The name of the member that is created is Qyydddhhmm, where yy is the year, ddd is the date in Julian format, and hhmm is the time (hours and minutes) when the performance monitor job is started.

member-name: Specify the name of the member that is used by the system. If a member does not exist, the system creates one with the specified name.

LIB

Specifies the library where the database files for performance data are collected. Each file that is not found in the specified library is automatically created by the system in that library.

The name of the database file can be qualified by one of the following library values:

QPFRDATA: The data is located in the IBM-supplied performance data library, QPFRDATA.

library-name: Specify the name of the library to be searched.

TEXT

Specifies text that briefly describes the database member receiving the performance data. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Note: This text appears for each member within the multiple files, associated with a measurement period.

***SAME:** The value does not change.

***GEN:** If the member is being created by the system, the text is 'Performance data collected dddddd at hhmm', where dddddd is the system date (in the format specified by system value (QDATFMT), and hhmm is the current time in hours and minutes. If the member already exists, no change is made.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

INTERVAL

Specifies the time interval (in minutes) between each collection of system performance data.

15: The default data collection interval value is 15 minutes.

number-of-minutes: Specify a collection interval value ranging from 5 through 60 minutes.

ENDTYPE

Specifies when the performance monitor stops collecting trace data.

***ELAPSED:** Data collection stops after the time specified in the hour and minute parameters has elapsed. Once the performance monitor has been started, changing the system clock does not affect the stop time.

***TIME:** Data collection stops at the time specified in the day, hour, and minute parameters.

***NOMAX:** Data collection does not stop until either the End Performance Monitor (ENDPFRMON) command is issued, or any of the database files to which the collected data is being sent becomes full.

The trace data collection duration is limited by the internal trace table size. Its duration is also dependent on the level of activity taking place on the system. If trace is being requested (by specifying TRACE(*ALL)), the total data collection period cannot exceed 12 hours. The trace function stops when the trace table is full, but the monitor continues to collect interval data.

DAY

Specifies, if ENDTYPE(*ELAPSED) is specified, the number of days during which to collect data. Specifies, if ENDTYPE(*TIME) is specified, the number of days from the current day until collection ends. If ENDTYPE(*NOMAX) is specified, this parameter has no meaning and is ignored.

0: Information is collected for 0 full days. Information can be collected for less than one full day using the HOUR and MINUTE parameters.

number-of-days: Specify the number of days. Valid values range from 0 through 7 days.

HOUR

Specifies, if ENDTYPE(*ELAPSED) is specified, the number of hours to collect data. Specifies, if ENDTYPE(*TIME) is specified, the hour of the day when data collection ends.

If ENDTYPE(*NOMAX) is specified, this parameter has no meaning and is ignored.

2: If ENDTYPE(*ELAPSED) is specified, data is collected for 2 hours. If ENDTYPE(*TIME) is specified, data collection ends at 2:00 a.m.

number-of-hours: Specify the number of hours. If ENDTYPE(*ELAPSED) is specified, valid values range from 0 through 999. If ENDTYPE(*TIME) is specified, valid values range from 0 through 23.

MINUTE

Specifies the number of minutes during which data is collected if ENDTYPE(*ELAPSED) is specified.

Specifies the minute of the specified hour when data collection ends if ENDTYPE(*TIME) is specified.

If ENDTYPE(*NOMAX) is specified, this parameter is ignored.

0: The value of 0 minutes is used.

number-of-minutes: Specify, if ENDTYPE(*ELAPSED) is specified, the number of minutes data is collected. A value ranging from 0 through 99 minutes can be specified.

If ENDTYPE(*TIME) is specified, specify the minute of the hour when data collection ends. A value ranging from 0 through 59 can be specified.

DATA

Specifies the type of data collected.

***ALL:** All data, including system data and communications data, is collected.

***SYS:** Only system data is collected.

TRACE

Specifies the type of internal trace data collection that is started.

Warning: Only one trace can be active in the operating system. If *ALL is specified on the TRACE parameter while another trace is running, the other trace is canceled and its data collection is lost.

***NONE:** No trace data collection is started.

***ALL:** Collection of all types of internal trace data that contain performance related information is started. If another trace is running at the same time this command specifying TRACE(*ALL) is issued, the other trace is canceled.

DMPTRC

Specifies whether the trace data is dumped to a database file when the data collection ends.

***YES:** The trace data is dumped to a database file when the data collection ends.

***NO:** The trace data is not dumped to a database file when the data collection ends. It can be dumped later by using the Dump Trace (DMPTRC) command.

JOBTRCITV

Specifies the time (in seconds of CPU operation) between each collection of the job trace data.

0.5: A time slice quantum interval value of 0.5 seconds is used.

number-of-seconds: Specify a time slice quantum value ranging from 0.5 through 9.9 seconds.

JOBTYPE

Specifies the types of jobs for which trace data is being collected for use in the batch job trace report. A maximum of 11 of the following job types can be traced, or the single value *NONE can be specified.

Note: The value *DFT includes the values *ASJ, *BCH, *EVK, *MRT, *PDJ, and *PJ. The value *BCH includes the values *EVK, *MRT, *PDJ, and *PJ.

***DFT:** Batch and autostart job types are traced.

***ASJ:** Autostart job types are traced.

***BCH:** Batch job types are traced.

***EVK:** Jobs started (evoked) by a procedure start request are traced.

***INT:** Interactive job types are traced.

***MRT:** Multiple requester terminal job types are traced.

***RDR:** Reader job types are traced.

***SBS:** Subsystem monitor job types are traced.

***SYS:** System job types are traced.

***WTR:** Writer job types are traced.

***PDJ:** Print driver job types are traced.

***PJ:** Prestart job types are traced.

Single Value

***NONE:** No job types are traced.

LRSPTIME

Specifies the local work station response time categories. The performance monitor keeps track of interactive response times for each local work station attached to a controller that supports collecting response time data. The response times are grouped into five categories and this parameter allows definition of the categories. Each value specified must be a three-position decimal number, with one decimal position.

***SYS:** The system response categories are:

- 0-1 seconds
- 1-2 seconds
- 2-4 seconds
- 4-8 seconds
- Longer than 8 seconds

Element 1: The First Boundary

boundary1: Specify the first response time boundary. Responses falling between zero and this boundary value are counted in the first response time category. This is a DEC(3,1) variable.

Element 2: The Second Boundary

boundary2: Specify the second response time boundary. Responses falling between the first boundary value and this boundary value are counted in the second response time category. This is a DEC(3,1) variable.

Element 3: The Third Boundary

boundary3: Specify the third response time boundary. Responses falling between the second boundary value

and this boundary value are counted in the third response time category. This is a DEC(3,1) variable.

Element 4: The Fourth Boundary

boundary4: Specify the fourth response time boundary. Responses falling between the third boundary value and this boundary value are counted in the fourth response time category. This is a DEC(3,1) variable.

Response times greater than the fourth response time category are counted in the fifth response time category.

RRSPTIME

Specifies the remote work station response time categories. The performance monitor keeps track of interactive response times for each remote work station attached to a controller that supports the collection of response time information. The response times are grouped into five categories; this parameter defines each category. Each value must contain a three-position decimal number, with one decimal position.

***NONE:** Remote work station response time is not collected.

***SYS:** The system response categories are:

- 0-1 seconds
- 1-2 seconds
- 2-4 seconds
- 4-8 seconds
- Longer than 8 seconds

Element 1: First Boundary

boundary1: Specify the first response time boundary. All responses falling between zero and this boundary value are counted in the first response time category. This is a DEC(3,1) variable.

Element 2: Second Boundary

boundary2: Specify the second response time boundary. All responses falling between the first boundary value and this boundary value are counted in the second response time category. This is a DEC(3,1) variable.

Element 3: Third Boundary

boundary3: Specify the third response time boundary. All responses falling between the second boundary value and this boundary value are counted in the third response time category. This is a DEC(3,1) variable.

Element 4: Fourth Boundary

boundary4: Specify the fourth response time boundary. All responses falling between the third boundary value and this boundary value are counted in the fourth response time category. This is a DEC(3,1) variable. All response times greater than this value are counted in the fifth response time category.

JOBQ

Specifies the qualified name of the job queue on which this job is placed.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QCTL: The IBM-supplied controlling subsystem, QCTL, is used.

job-queue-name: Specify the name of the job queue.

MSGQ

Specifies the qualified name of the message queue to which the performance monitor sends messages, in addition to the system operator's message queue (QSYS/QSYSOPR). Messages sent to the message queue communicate status information relating to performance monitor startup, ending, and problems.

***NONE:** Messages are sent to the system operator message queue only.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue to which messages are sent.

EXITPGM

Specifies the user-written exit program that is called to process the performance data collected by this command.

***NONE:** No exit program is specified.

The name of the exit program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

user-label-program-name: Specify the user label name of the exit program to be used.

HLICDATA

Specifies the type of horizontal licensed internal code (HLIC) data to be collected.

***NONE:** No HLIC data is collected.

***CPU:** Internal processing unit (CPU)-related HLIC data is collected.

Note: Do not select this option unless instructed to do so by an IBM representative.

Examples

Example 1: Ending After Seven Days

```
STRPFRMON  INTERVAL(30)  ENDTYPE(*TIME)  DAY(7) HOUR(16)
  MINUTE(30)
```

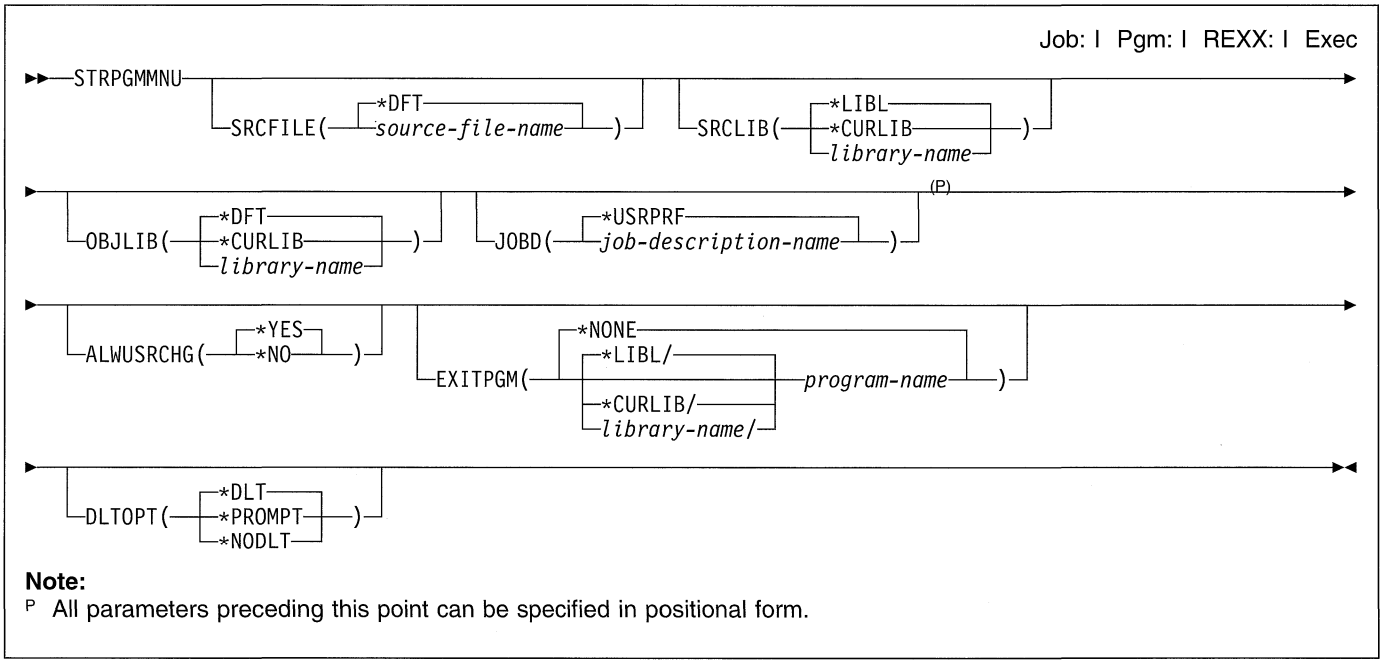
In this example, the performance monitor is submitted to the QCTL job queue in the QSYS library. The performance monitor generates the member name (based upon the time that the performance monitor is started from the job queue), puts performance data into the performance database files in library QPFRDATA every 30 minutes, and ends 7 days from the current day at 4:30 in the afternoon.

Example 2: Submitting a Job With No End Time

```
STRPFRMON  MBR(JULY12)  ENDTYPE(*NOMAX)  JOBQ(QGPL/QBATCH
  MSGQ(MYMSGQ)
```

In this example, the performance monitor is submitted to the QBATCH job queue in the QGPL library. Performance data is collected in the performance database files QPFRDATA and the member name is JULY12. The performance does not have an end time, but it can be ended by using the End Performance Monitor (ENDPFRMON) command. Messages indicating the status of the performance monitor are sent to the MYMSGQ message queue (in addition to the QSYSOPR message queue).

STRPGMMNU (Start Programmer Menu) Command



Note:

| (P) All parameters preceding this point can be specified in positional form.

Purpose

The Start Programmer Menu (STRPGMMNU) command shows the Programmer Menu. This command can be used instead of the CALL QPGMMENU function, and allows the user to pass parameters to specify and control the data which appears in the associated fields on the programmer menu.

Notes:

1. A user exit program can be called instead of submitting a job when option 3 is selected.
2. The first four parameters control the defaults that appear when the menu is first displayed.

More information about using the Programmer Menu is in the *CL Programmer's Guide*.

Optional Parameters

SRCFILE

Specifies the name of an existing source file that contains source file members being updated or to which new members are being added.

***DFT:** This is the default for the TYPE of source file being specified on the menu. This field is blank when shown on the display.

source-file-name: Specify the name of the source file being updated.

SRCLIB

Specifies the qualified name of the library that is searched for the source file.

The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

OBJLIB

Specifies the name of the library, which was created with option 3, that receives objects.

***DFT:** The library used is dependent on the menu option selected. If this value is specified, the object library field will be blank.

***CURLIB:** The current library for the job contains or receives the object.

library-name: Specify the name of the library that receives the created object.

JOBD

Specifies the job description used to submit jobs for batch processing. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***USRPRF:** The job description included in the user profile is used for the job.

job-description-name: Specify the name of the job description that is used for the job. (The job description is found through the library list being used by the job.)

ALWUSRCHG

Specifies whether the menu screen fields, specified in the previous parameters in this command, can be changed by the user.

***YES:** The values on the display can be changed.

***NO:** The display fields cannot be changed. This option is intended to minimize errors, and is not considered a security function.

EXITPGM

Specifies the qualified name of a user-written program that is called as an exit program in place of submitting a batch job when menu option 3 is selected. When the exit program is called, it receives parameters that are sent by the Programmer Menu. (Refer to the examples at the end of this command description.) More information about the EXITPGM parameter is in the *CL Programmer's Guide*.

***NONE:** No user-written program is called; a batch job is submitted. When *NONE is specified, DLTOPT(*DLT) must be specified.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of the program called when option 3 is selected, instead of submitting the create command as a batch job. When a value is specified for this parameter, the text that appears on the menu for option 3 shows the name and library of the exit program.

DLTOPT

Specifies the action that is taken when (1) a program name is specified for the EXITPGM parameter, (2) option 3 of the Programmer Menu is selected, and (3) an object of the name and type being created exists in the library specified by the OBJLIB field of the menu. Regardless of the value specified, the system passes a parameter (from among the parameters passed from the Programmer Menu) to the exit program that specifies whether the object exists.

***DLT:** This value must be specified if *NONE is specified for the EXITPGM parameter. If an EXITPGM is specified, and the object specified to be created with option 3 exists, and the Enter key is pressed, a message is shown; press the F11 key to proceed. When the F11 key is pressed, the system deletes or replaces the object before calling the program specified by the exit program. This is the normal Programmer Menu function when an EXITPGM is not specified.

Notes:

1. When the *DLT value is specified, the object is deleted or replaced before the job is submitted or the user exit program is called.
2. If the source type is one of the following, the object is replaced rather than deleted (an exit program must be used to delete instead of replace):

```
BAS       C         CBL
CBL36     CLP       DSPF
DSPF36    FTN       ICFF
MNU36     MSGF36    PAS
PLI       PRTF      RPG
RPG36     RPT36
```

***PROMPT:** The system does not delete or replace the object, but the user is prompted for approval to delete the object. If the object exists, and the Enter key is pressed, a message is displayed. Press the F11 key to proceed; the system does not delete the object. The user confirms whether the object is deleted or replaced, yet the deletion is still controlled by the exit program.

***NODLT:** The user exit program is called regardless of the presence of the object.

Table 75 shows the actions taken by the system when the exit program is called. The value passed to the exit program is a character variable of length 1 that contains the character value shown. Note that a value other than 0 indicates that the object does not exist.

Table 75. Actions Taken by the System When an Exit Program Is Called

DLTOBJ Specified	F11 Required	Object Deleted	Value Passed to Exit Program
If object existed when option 3 was selected			
*DLT	Yes	Yes	1
*PROMPT	Yes	No	0
*NODLT	No	No	0
If object did not exist when option 3 was selected			
*DLT	No	---	2
*PROMPT	No	---	2
*NODLT	No	---	2

Examples

Example 1: Displaying Programmer Menu

```
STRPGMMNU
```

This command displays the Programmer Menu with defaults for all parameters. This has the same result as entering CALL QPGMMENU.

Example 2: Preventing Values from Being Changed

```
STRPGMMNU  SRCFILE(YOURFILE)  SRCLIB(YOURLIB)
  OBJLIB(YOURLIB)  JOB(YOURJOBD)  ALWUSRCHG(*NO)
```

This command prevents the values on the menu from being changed from those specified on the command.

Example 3: Calling an Exit Program

```
STRPGMMNU  EXITPGM(OPT3PGM)  DLTOPT(*PROMPT)
```

This command calls user exit program OPT3PGM instead of submitting a batch job when option 3 is specified. If the object already exists, DLTOPT(*PROMPT) requires the user to press the F11 key; however, the object is not deleted.

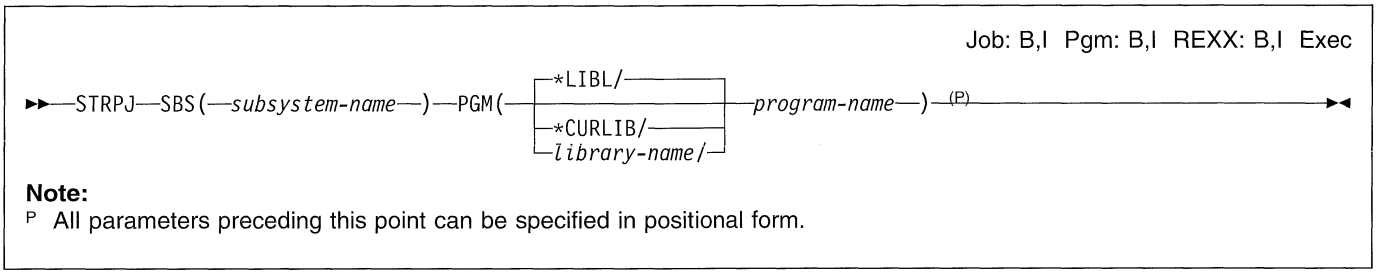
Example 4: Receiving Parameters

The following portion of a CL program is an example of how these parameters would be received by a user exit program. If the specified type is one of those listed, the object is not deleted. The create command with REPLACE(*YES) specified is passed to the exit program. The value passed to the exit program is 0.

```
PGM PARM(&OPTION &PARM &TYPE &PARM2
  &SRCFILE &SRCLIB
  &OBJLIB &JOBD &RQSLEN &RQSDTA512
  &F4 &F11 &EXIST)

/* The following values are passed in exactly as  */
/* they appear on the Programmer Menu.            */
    DCL VAR(&OPTION)  TYPE(*CHAR)  LEN(2)
    DCL VAR(&PARM)    TYPE(*CHAR)  LEN(10)
    DCL VAR(&TYPE)    TYPE(*CHAR)  LEN(10)
    DCL VAR(&PARM2)   TYPE(*CHAR)  LEN(21)
    DCL VAR(&SRCFILE) TYPE(*CHAR)  LEN(10)
    DCL VAR(&SRCLIB)  TYPE(*CHAR)  LEN(10)
    DCL VAR(&OBJLIB)  TYPE(*CHAR)  LEN(10)
    DCL VAR(&JOBD)    TYPE(*CHAR)  LEN(10)

/* The following values are derived by QPGMMENU     */
/* from the information entered to the above fields */
/* and the F keys.                                  */

   /* NUMBER OF BYTES OF REQUEST DATA */
   DCL  VAR(&RQSLEN)  TYPE(*DEC)   LEN(3 0)

   /* DATA FOR RRQSDTA PARAMETER OF SBMJOB COMMAND. */
   DCL  VAR(&RQSDTA512)  TYPE(*CHAR)   LEN(512)

   /* F4 WAS PRESSED, '1', OTHERWISE '0'.   */
   DCL  VAR(&F4)    TYPE(*CHAR)   LEN(1)

   /* F11 WAS PRESSED, '1', OTHERWISE '0'.     */
   DCL  VAR(&F11)   TYPE(*CHAR)   LEN(1)

   /* OBJECT EXISTS- '0 OBJECT WAS DELETED- '1'
   OR OBJECT DID NOT EXIST -'2'*/
   DCL  VAR(&EXIST)  TYPE(*CHAR)   LEN(1)
```

Additional information, along with examples of the STRPGMMNU command with the EXITPGM parameter, is in the *CL Programmer's Guide*

STRPJ (Start Prestart Jobs) Command



Purpose

The Start Prestart Jobs (STRPJ) command starts jobs for a prestart job entry in an active subsystem when there are no currently active prestart jobs for the prestart job entry. This command is valid after an ENDPJ command is complete or when all prestart jobs have been ended by the system due to an error. This command is also valid after a STRSBS (Start Subsystem) command where the prestart job entry indicates the prestart jobs should not be started when the subsystem is started. The number of jobs started is determined by the INLJOBS value on the prestart job entry and is limited by the MAXJOBS value for the subsystem.

Restriction: This command is restricted to a user with job control special authority. The user must also have *USE authority to the subsystem description.

Required Parameters

SBS

Specifies the name of the active subsystem that contains the prestart job entry. Specify the name of the active subsystem.

PGM

Specifies the qualified name of the program for the prestart job entry. This is also the name of the program that the prestart job runs. This program name is used to match an incoming program start request with an available prestart job.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

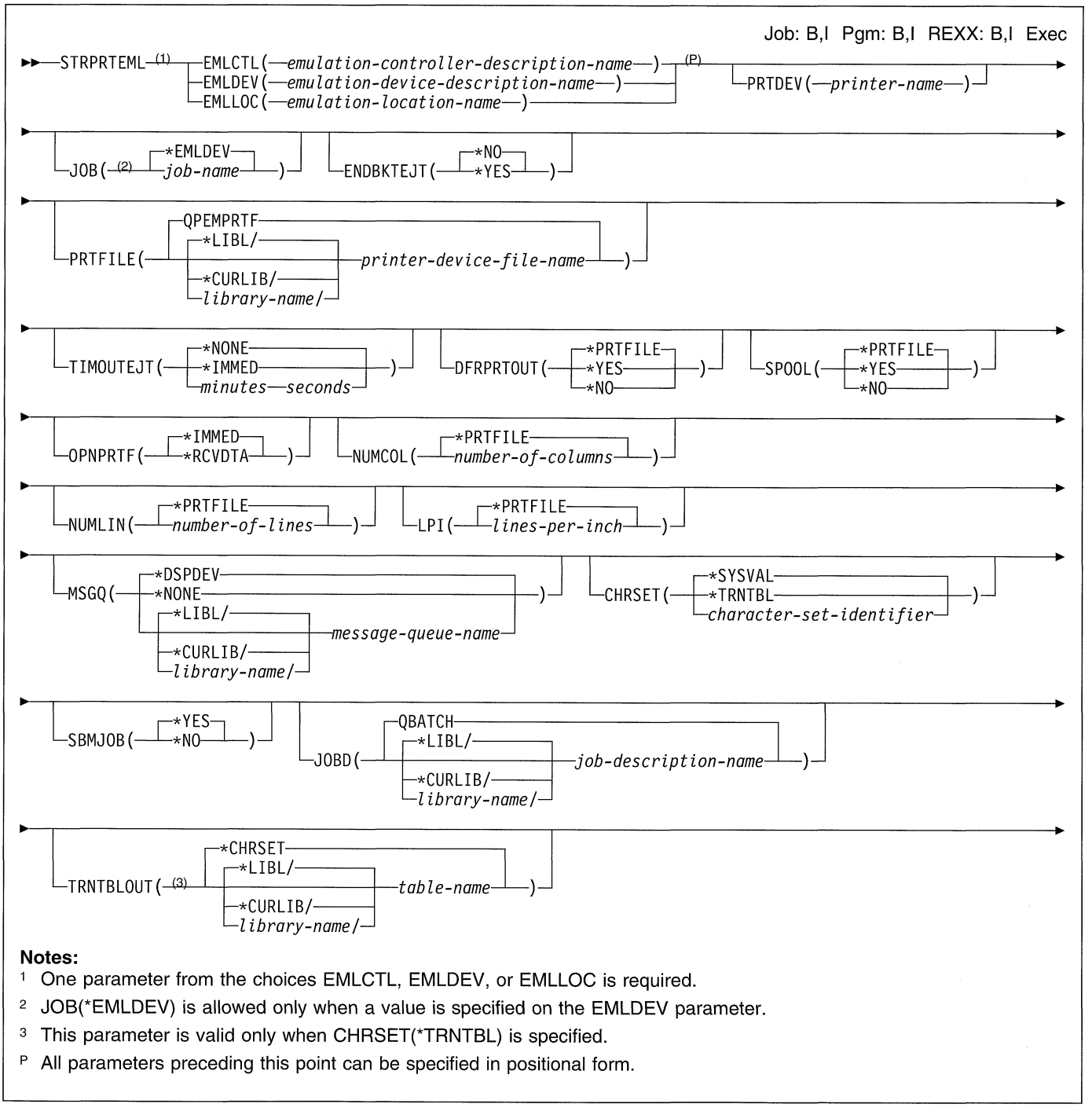
program-name: Specify the name of the program for the prestart job entry.

Example

```
STRPJ  SBS(SBS1)  PGM(PJLIB/PJPGM)
```

This command starts prestart jobs for prestart job entry PJPGM in subsystem SBS1. Subsystem SBS1 must be active when this command is issued. The number of jobs started is the number specified in the INLJOBS value of prestart job entry PJPGM. The subsystem starts program PJPGM in library PJLIB.

STRPRTEML (Start Printer Emulation) Command



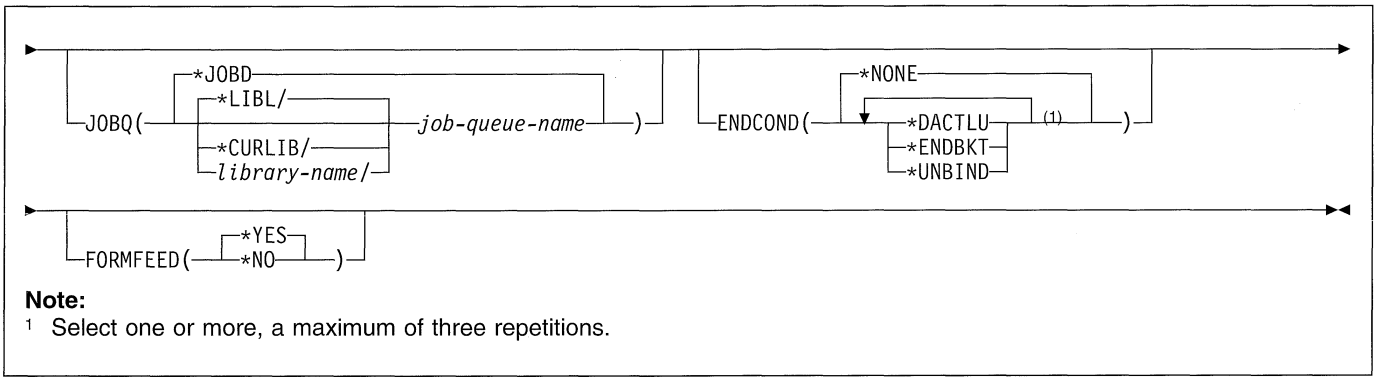
Notes:

1 One parameter from the choices EMLCTL, EMLDEV, or EMLLOC is required.

2 JOB(*EMLDEV) is allowed only when a value is specified on the EMLDEV parameter.

3 This parameter is valid only when CHRSET(*TRNTBL) is specified.

P All parameters preceding this point can be specified in positional form.



Note:

1 Select one or more, a maximum of three repetitions.

Purpose

The Start Printer Emulation (STRPRTEML) command is used to start 3270 printer emulation using a binary synchronous communications (BSC) or Systems Network Architecture (SNA) emulation printer device and a printer device file. The STRPRTEML command is used to print host system (System/370 type) information on an AS/400 system. It is used when the user is working on an AS/400 system and the information is on a System/370 type system.

More information is in the *3270 Device Emulation Guide*.

Required Parameters

EMLCTL

Specifies the name of a BSC controller description for which (APPTYPE(*EML)) is specified, or an SNA controller description type of HOST that has attached 3270 emulation printer device descriptions. If this parameter is specified, the printer emulation job uses a 3270 printer device attached to this controller description. The requester must be authorized to the controller and to the device, and the device must be available.

EMLDEV

Specifies the name of a BSC or an SNA printer emulation device (EMLDEV(3284, 3286, 3287, 3288, or 3289)) that is used by the printer emulation job to do a type 3270 printer emulation. The user must be authorized for the device, and the device must be available.

EMLLOC

Specifies the location name associated with this session. This name is defined during configuration and refers to the remote location where communication takes place.

Optional Parameters

PRTDEV

Specifies the name of the printer used with this printer device file to create the printed output. If the printer output for this printer is not being spooled, the printer must not be in use.

JOB

Specifies a job name for the printer emulation job. If the EMLDEV parameter is not specified, and a batch job is to be submitted (SMBJOB(*YES)), a job name must be specified. If SBMJOB(*NO) is specified, this parameter is ignored. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***EMLDEV:** The job name is the same as the print emulation device name.

job-name: Specify the name to be assigned to the job.

ENDBKTEJT

Specifies whether SNA printer emulation forces out the emulation output when an SNA End Bracket (EB) is received from the host system. Emulation output is forced out by closing and then reopening the emulation printer file specified in the PRTFILE parameter. When the emulation printer output is ejected, a page eject is performed. This parameter uses the default value of *NO for BSC printer emulation.

***NO:** The emulation output is not forced out when SNA printer emulation receives an End Bracket.

***YES:** The emulation output is forced out when SNA printer emulation receives an End Bracket. This is done only if the open printer file contains host system data.

PRTFILE

Specifies the printer device file that prints data received from the host system. The printer device file can be spooled or not spooled. If the files from the host system are to be printed on the printer device, the user must specify the printer device name in the printer file for files with SPOOL(*NO) specified. For files with SPOOL(*YES) specified, the user must start a writer to the output queue for this job, and print to the associated printer device.

QPEMPRTF: The standard printer file (which specifies SPOOL(*YES)) shipped with the emulation utility is used as the printer device file.

The name of the printer device file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

printer-device-file-name: Specify the qualified name of a user-defined printer device file.

TIMOUTEJT

Specifies whether printer emulation forces out the emulation output when a timeout has occurred while waiting for host system data. Emulation output is forced out by closing and then reopening the emulation printer file specified in the PRTFILE parameter. When the emulation printer output is ejected, a page eject is performed.

***NONE:** The emulation output is not forced out based on a specified timeout period.

***IMMED:** The emulation output is forced out immediately.

Element 1: Minutes

minutes: Specify the number of minutes in the time-out wait interval. Valid values range from 0 through 99.

Element 2: Seconds

0: No seconds are specified for the time-out wait interval.

seconds: Specify the number of seconds in the time-out wait interval. Valid values range from 0 through 59.

DFRPRTOUT

Specifies whether spooled output is printed immediately or is delayed (deferred).

***PRTFILE:** The printer file determines whether the spooled output is printed immediately.

***YES:** Printing of spooled output is delayed until the printer is signed off of device emulation, or until a move operation is done.

***NO:** Spooled output can be printed before the spooled file is closed. The printed output does not contain all the data sent by the host system until the spool file is closed. If the printer is not using spooling, this parameter is ignored.

Note: Once the printer starts printing output from 3270 device emulation, spooled output from other jobs sharing the printer does not print until the spooled file that is currently printing is complete.

SPOOL

Specifies whether the output data for the printer device file is spooled.

***PRTFILE:** The printer file selects whether spooling is done.

***YES:** The data is spooled.

***NO:** The data is not spooled; it is sent directly to the device and is printed as the output becomes available.

OPNPRTF

Specifies when the printer file is opened during the SNA 3270 printer emulation session. If the printer data is not spooled, then the printer is allocated to this job when the printer file is opened. If the printer data is spooled, then the spool writer is allocated to this job after the printer file is opened, depending on the value of the DFRPRTOUT parameter.

This parameter is not allowed if specified for BSC 3270 printer emulation.

***IMMED:** The printer file is opened immediately after the 3270 printer emulation session is started.

***RCVDTA:** The printer file is opened after receiving print data from the host system.

NUMCOL

Specifies the number of columns in a line when creating printed output.

***PRTFILE:** The printer file contains the number of columns per line. This value is used if OPNPRTF(*IMMED) or OPNPRTF(*RCVDTA) is specified and the maximum print positions (MPP) value is not sent from the host system. Otherwise, the MPP value sent from the host system is used.

number-of-columns: Specify the number of columns per line in the printed output. Valid values range from 1 through 378.

NUMLIN

Specifies the number of lines per page when creating the printed output.

***PRTFILE:** The printer file contains the number of lines per page. This value is used if OPNPRTF(*IMMED) or *RCVDTA is specified and the maximum page length (MPL) value is not sent from the host system. Otherwise, the MPL value sent from the host system is used.

number-of-lines: Specify the number of lines per page in printed output. Valid values range from 1 through 255.

LPI

Specifies the number of lines per inch when creating the printed output.

***PRTFILE:** The printer file contains the number of lines per inch. This value is used if OPNPRTF(*IMMED) or OPNPRTF(*RCVDTA) is specified, and the set line density (SDL) value is not sent from the host system. Otherwise, the SDL value sent from the host system is used.

lines-per-inch: Specify the number of lines per inch in the printed output. Valid values are 3, 4, 5, 6, 7, 7.5, 8 and 9. Values 3, 5, 7, and 7.5 are valid only for double-byte character set (DBCS-capable) printer devices.

MSGQ

Specifies the qualified name of the message queue to which messages are sent.

***DSPDEV:** The current display device message queue is used. If this value is specified from a batch job, no messages are sent to message queues other than the job log for the printer emulation job.

***NONE:** No messages are sent to message queues other than the job log for the printer emulation job.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue where messages are sent when the printer emulation job is run.

CHRSET

Specifies the language character set that is used by the printer being emulated.

***SYSVAL:** The value specified in the system value QKBDTYPE is used.

***TRNTBL:** User-defined translation tables are used. The character translation is defined in the translation table specified by the TRNTBLOUT parameter.

character-set-identifier: Specify the identifier of the character set used by the printer being emulated. Three-character identifiers by language or country are listed in the Keyboard Mapping Table.

Table 76. Keyboard Mapping Table

Language/Country	Identifier	ASCII Device Group
Arabic X/Basic	CLB	D
Austria/Germany	AGB	A, B
Austria/Germany Multinational	AGI	A, B
Belgium Multinational	BLI	B
Brazilian Portuguese	BRB	
Canadian French	CAB	A, B
Canadian French Multinational	CAI	A, B
Cyrillic	CYB	
Denmark	DMB	B
Denmark Multinational	DMI	B
Finland/Sweden	FNB	B
Finland/Sweden Multinational	FNI	B
France (Azerty)	FAB	A, B
France (Azerty) Multinational	FAI	A, B
France (Qwerty)	FQB	
France (Qwerty) Multinational	FQI	
Greece	GNB	
Greece	GKB	
Hebrew	NCB	D
Iceland	ICB	
Iceland Multinational	ICI	
International	INB	
International Multinational	INI	
Italy	ITB	A, B
Italy Multinational	ITI	A, B
Japan English	JEB	
Japan English Multinational	JEI	
Japan Kanji (For PS*/55 and 5295 display stations)	JKB	
Japan United States Basic	JUB	
Japan Katakana (For 5251, 5291, 5292, and 3180 Katakana display stations)	KAB	
Korea	KOB	
Latin-2/ROECE	ROB	
Netherlands	NEB	
Netherlands Multinational	NEI	
Norway	NWB	B
Norway Multinational	NWI	B
Portugal	PRB	B
Portugal Multinational	PRI	B
Simplified Chinese	RCB	
Spain	SPB	B
Spain Multinational	SPI	B
Spanish Speaking	SSB	B
Spanish Speaking Multinational	SSI	B
Sweden	SWB	B
Sweden Multinational	SWI	B
Switzerland/French Multinational	SFI	B
Switzerland/German Multinational	SGI	B
Thai	THB	
Traditional Chinese	TAB	
Turkey	TKB	
United Kingdom	UKB	A, B
United Kingdom Multinational	UKI	A, B
United States/Canada	USB	A, B, C
United States/Canada Multinational	USI	A, B, C
Languages of the former Yugoslavia	YGI	

Note: For example, CHRSET(USB) indicates that the basic United States character set is used.

SBMJOB

Specifies whether the printer emulation is done as a separate job or as part of this job.

***YES:** A specific job is submitted to do the printer emulation. The job attributes and job queue are determined from the job description (JOBD parameter). The job uses the user's profile.

***NO:** Printer emulation is done in the current job. This allows the user to specify an Override Printer File (OVRPRTF) command before the printer emulation request to override the printer file. The user can also follow the printer emulation request with any other system request, such as copy spooled file to database

file. SBMJOB(*NO) can be used only in a batch job; however, it is allowed in an interactive job.

JOBD

Specifies the job description for the job that is being submitted for 3270 printer emulation. If SBMJOB(*NO) is specified, this parameter is ignored. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

QBATCH: The job description QBATCH is to be used for the job.

The name of the job description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-description-name: Specify the qualified name of the job description associated with the job.

TRNTBLOUT

Specifies the outgoing translation table used to translate characters sent from the host system to the remote 3270 emulation.

***CHRSET:** Specifies that translation is done when data is sent from the host system using the character set specified in the CHRSET parameter.

The name of the translation table can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

table-name: Specify the qualified name and library of the table that is used for outgoing translation.

JOBQ

Specifies the qualified name of the job queue on which this job is placed.

***JOBD:** The submitted job is placed in the job queue associated with the job description specified in the job description (JOBD) parameter.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-queue-name: Specify the qualified name of the job queue on which the submitted job is placed.

ENDCOND

Specifies additional ways in which the SNA 3270 printer emulation session can end.

This parameter is not allowed if specified for BSC 3270 printer emulation.

***NONE:** No additional ways to end 3270 printer emulation are requested.

***DACTLU:** The 3270 printer emulation session ends if it receives an SNA DACTLU from the host system.

***ENDBKT:** The 3270 printer emulation session ends if it receives an SNA end bracket from the host system.

Note: This end condition is used only when printing one host system file for the duration of the session. An end bracket can occur after printing the first file, and the 3270 session ends before a second file can print.

***UNBIND:** The 3270 emulation session ends if it receives an SNA UNBIND from the host system.

Note: This end condition is used only when printing one host system file for the duration of the session. An UNBIND can occur after printing the first file, and the 3270 session ends before a second file can print.

FORMFEED

Specifies whether to acknowledge a form-feed instruction located in the first character position of the first print line for a 3270 Information Display System data-stream compatibility (DSC) printer.

This parameter is ignored for an SNA character string (SCS) printer.

***YES:** The form-feed instruction is acknowledged. The print position advances to a new page.

***NO:** The form-feed instruction is ignored. The print position does not advance to a new page.

Examples

Example 1: Printing Data to Standard Emulation Printer File

```
STRPRTEML  EMLDEV(HOSTPRT4)
```

This command starts a batch job by accepting data from the HOSTPRT4 device and prints the data to the standard emulation printer file (QPEMPRTF). The job is named HOSTPRT4 and runs until the job is canceled. Messages are sent to the current work station message queue.

Example 2: Printer Emulation Done in Current Job

```
STRPRTEML  EMLDEV(HOSTPRT5)  SBMJOB(*NO)
```

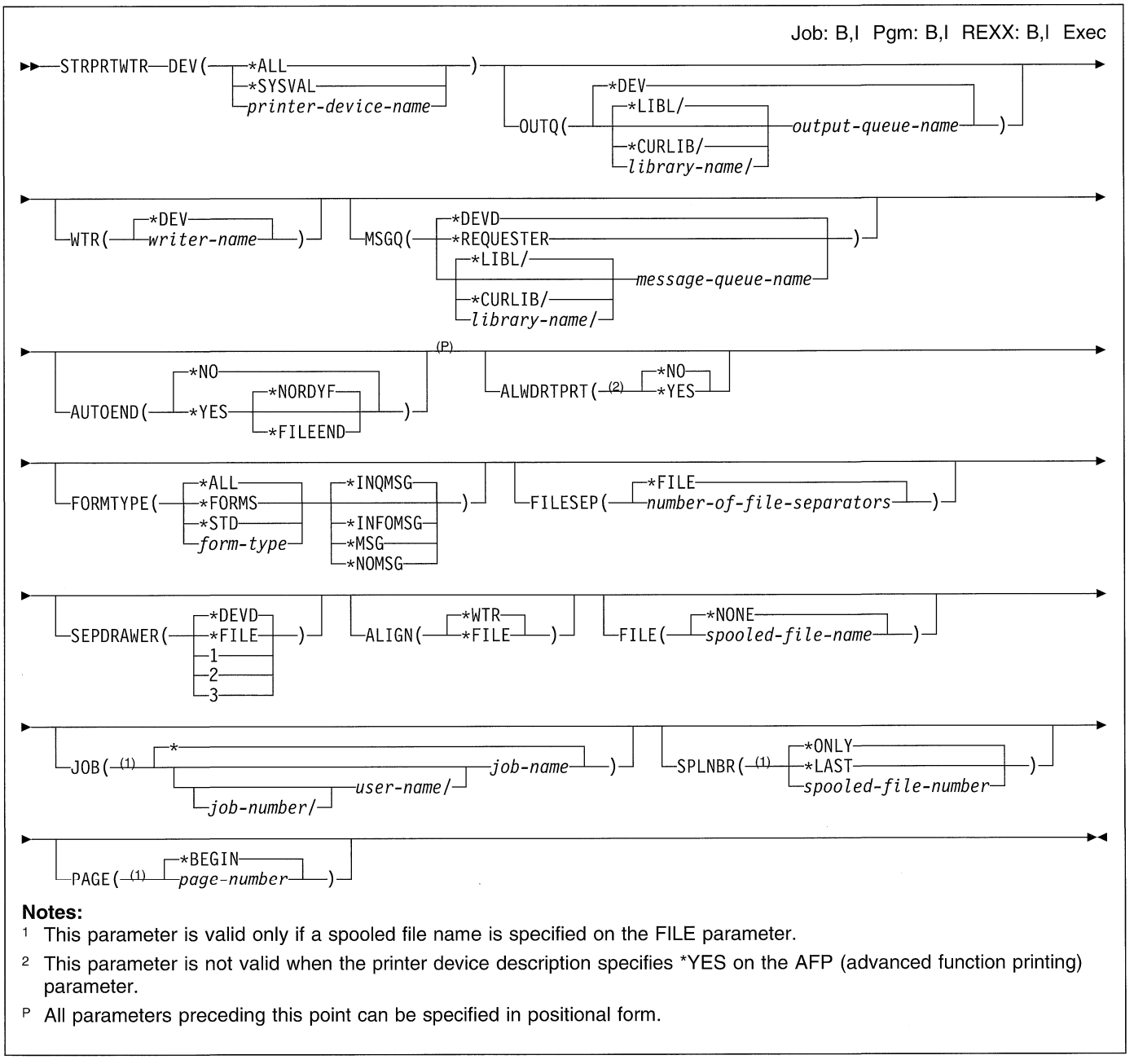
This command does printer emulation in the current job by accepting data from the HOSTPRT5 device, and writing the data to the standard emulation printer device file (QPEMPRTF). The request is active until it ends through the End Printer Emulation (ENDPRTEML) command, or until the job is canceled.

Example 3: Printing Output Immediately

```
STRPRTEML  EMLCTL(EMLCTL1)  TIMOUTEJT(10)  DFRPRTOUT(*NO)
NUMLIN(96)
```

This command starts a batch job by accepting data from the device and printing the data in printer file QPEMPRTF. If a timeout of 10 minutes occurs, printer emulation forces out the emulation output. The output prints immediately; the maximum number of lines per page is 96.

STRPRTWTR (Start Printer Writer) Command



Notes:

1 This parameter is valid only if a spooled file name is specified on the FILE parameter.

2 This parameter is not valid when the printer device description specifies *YES on the AFP (advanced function printing) parameter.

P All parameters preceding this point can be specified in positional form.

Purpose

The Start Printer Writer (STRPRTWTR) command starts a spooling writer that moves spooled files from an output queue to a specified printer. The writer, which is a system job, takes spooled files from an output queue and produces (writes) the output on the printer device. This command specifies the name of the printer, the names of the output and message queues used, and the name of the writer.

More than one writer can be active at the same time (as determined by the spooling subsystem description). Each writer must have a unique writer name, its own output queue, and its own device. A writer that has been started can be actively writing output or waiting for a file to be put on the output queue. Optionally, the writer can end automatically when it has processed all the files on the output queue. The writer can also be changed, held, or canceled if the Change Writer (CHGWTR), Hold Writer (HLDWTR), or End Writer (ENDWTR) command is used.

Because each writer runs independent of the job that started it, users can continue doing other work on the system after they have started a writer. The writer is owned by the user who issues the STRPRTWTR command.

Required Parameter

DEV

Specifies the name of the printer device used to print the spooled file. The device must be available for allocation before the writer can be started.

***ALL:** A printer writer is started for every printer configured on the system.

***SYSVAL:** A printer writer is started for the system default printer.

printer-device-name: Specify the name by which the printer device being started is identified.

Optional Parameters

OUTQ

Specifies the qualified name of the output queue.

***DEV:** The default output queue associated with the printer specified on the DEV parameter is used.

The name of the output queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

output-queue-name: Specify the name of the output queue from which the writer processes output files.

WTR

Specifies the name of the spooling writer being started. Each writer name must be unique.

***DEV:** The name of the writer is the same as that of the printer device specified on the DEV parameter.

writer-name: Specify the name by which the writer being started is identified.

MSGQ

Specifies the qualified name of the message queue to which messages created by the writer are sent.

***DEVD:** Messages are sent to the message queue specified in the device description of the device named on the DEV parameter.

***REQUESTER:** The messages are sent to the message queue of the user who started the writer. If this value is specified for a batch job, *DEVD is used.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the name of the message queue to which messages created by the writer are sent.

AUTOEND

Specifies whether the writer ends automatically.

***NO:** The writer does not end when the last available file has been removed from the output queue. It waits for another spooled file entry to be put on the queue.

Element 1: Ending a Writer Automatically

***YES:** The writer automatically ends after it has reached the state specified on Element 2 of this parameter (*NORDYF or *FILEEND).

Element 2: Ending a Writer with a Ready File

***NORDYF:** The writer automatically ends when there are no ready files (all the available files have been removed from the output queue).

***FILEEND:** The writer ends after it finishes processing one spooled file.

ALWDRTPRT

Specifies whether the printer writer allows files to be printed directly to the printer. A file printed directly to the printer is created by specifying SPOOL(*NO) for a printer file. When direct printing is allowed, the non-spooled printer file is printed immediately if the printer is available or, if the printer is busy, the non-spooled printer file waits until the printer is available. The maximum wait is the length of time specified on the WAITFILE parameter on the printer file, after which the job is automatically canceled. The user can cancel a non-spooled printer file only with an End Job (ENDJOB) command.

***NO:** The printer does not allow non-spooled printer files to be printed to the device.

***YES:** The printer can be used to print spooled and non-spooled output. See the Create, Change, or Override Printer File (CRTPRTF, CHGPRTF, or OVRPRTF) command to set the value of the WAITFILE parameter.

Note: Nonspooled files wait up to 30 seconds regardless of whether the value specified on the WAITFILE parameter is less than 30 seconds.

FORMTYPE

Specifies the form type selection values that govern which spooled files are produced by the writer. A file's form type is specified in the device file that produced the spooled file. This parameter specifies that only the files with this form type are processed now. Files with other form types are left on the output queue and are available when their forms types are specified.

When a writer is sent to a printer device, a form load message is shown on the user's display if:

- The printer device has not been configured since it was last varied on.
- The form type mounted on the printer device does not match the form type of the first file spooled to the printer.

If the previous conditions occur, the form load message instructs the user to load the appropriate form type in the printer device.

To change the type of form for which spooled files are being produced, use the CHGWTR command.

Note: The form load message is issued when the spooled file to be printed has a form type different from the form type of the last spooled file that was printed on the device. The value for the last form type printed is kept from the last STRPRTWTR, CHGWTR, or VRYCFG command issued.

Consider the following example:

1. The last spooled file printed on printer PRT01 had the form type *STD.
2. The user changes the form type on PRT01 to XYZ using the following command:

   ```
   CHGWTR  PRT01  FORMTYPE(XYZ)
   ```

3. No spooled file with the form type XYZ is printed on PRT01.
4. The user then sends a spooled file with the form type *STD to PRT01. The form load message is not issued, despite the intervening CHGWTR command, because the last spooled file printed on PRT01 had the same form type as the spooled file being printed.

   The form load message would be issued if a spooled file with the form type XYZ were actually printed on PRT01.

Element 1: Type of Form Designation

***ALL:** All form types are processed by the writer.

***FORMS:** Available files on the output queue with the same form type are processed as a group before the writer moves on to the next form type group. The writer first chooses the first available file on the queue. After the first file is complete, all files with the same form type are processed. The writer again chooses the first available file on the queue and repeats the process for that form type.

***STD:** Only files that specify the standard form type are selected.

form-type: Specify the form type of the spooled files being produced.

Element 2: Message Sending Options

***INQMSG:** An inquiry message is sent to the message queue when a spooled file has a form type that is different than the form type in the printer.

***INFOMSG:** An informational message is sent to the message queue when no spooled files requiring this form type remain in the output queue.

***MSG:** An inquiry message is sent to the message queue when a spooled file has a form type that is different than the form type in the printer and an informational message is sent when no spooled files requiring this form type remain in the output queue.

***NOMSG:** Neither an inquiry message nor an informational message is sent to the message queue.

FILESEP

Specifies how to control the number of file separator pages that are printed preceding each file.

***FILE:** The number of separators specified for each individual file is used.

number-of-file-separators: Specify the number of separator pages to print. Valid values range from 0 through 9. Whenever the user responds to the change form type message indicating that a new form type has been put on the printer, the writer issues a message inquiring how many file separator pages are printed with the new form type.

SEPDRAWER

Specifies which drawer is selected for printing separators.

***DEVD:** The value stored in the device description for the printer is used.

***FILE:** The separator pages are printed on paper from the same drawer as the rest of the spooled file.

1: The separator pages are printed from drawer 1.

2: The separator pages are printed from drawer 2.

3: The separator pages are printed from drawer 3.

ALIGN

Specifies how to control the forms alignment.

***WTR:** The writer keeps track of the output that is printed and issues a forms alignment message whenever it determines that alignment is needed.

***FILE:** The forms alignment message is issued for every file for which ALIGN(*YES) is specified. This option must be selected whenever the automatic forms alignment control provided by the writer does not provide the desired results.

FILE

Specifies the name of the first (or only) spooled file being processed by the spooling writer and printed. If several files are available on the output queue, the next file produced is the first one available that has the highest priority.

***NONE:** No spooled file name is specified. The first spooled file that becomes available on the output queue is processed first.

spooled-file-name: Specify the name of the spooled output file that is the first (or only) output file written to the printer.

JOB

Specifies the name of the job that created the spooled file. This parameter is valid only if a spooled file name is specified on the FILE parameter. If no job qualifier is given, all of the jobs currently in the system are searched for the job name.

A job identifier is a special value or a qualified name with up to three elements. For example:

```
*
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** The job from which this STRPRTWTR command was entered is the job that created the spooled file.

job-name: Specify the name of the job that created the spooled file.

user-name: Specify the name of the user of the job that created the spooled file.

job-number: Specify the number of the job that created the spooled file.

SPLNBR

Specifies the number of the spooled file being processed first. This parameter is valid only if a spooled file name is specified in the FILE parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ONLY:** One spooled file from the job has the specified file name. The number of the spooled file is not necessary. If *ONLY is specified and more than one spooled file has the specified file name, a message is sent.

***LAST:** The spooled file with the highest number and the specified file name is used.

spooled-file-number: Specify the number of the job's spooled file that is on the specified output queue and that is being processed first.

PAGE

Specifies the number of the first page to print of the first file specified on the FILE parameter.

***BEGIN:** Printing begins on the restart page of the spooled file. If the RESTART parameter value is *STRPAGE on the Change Spooled File Attributes (CHGSPLFA) command, then printing begins on the page specified on the PAGERANGE parameter, which is also on the CHGSPLFA command.

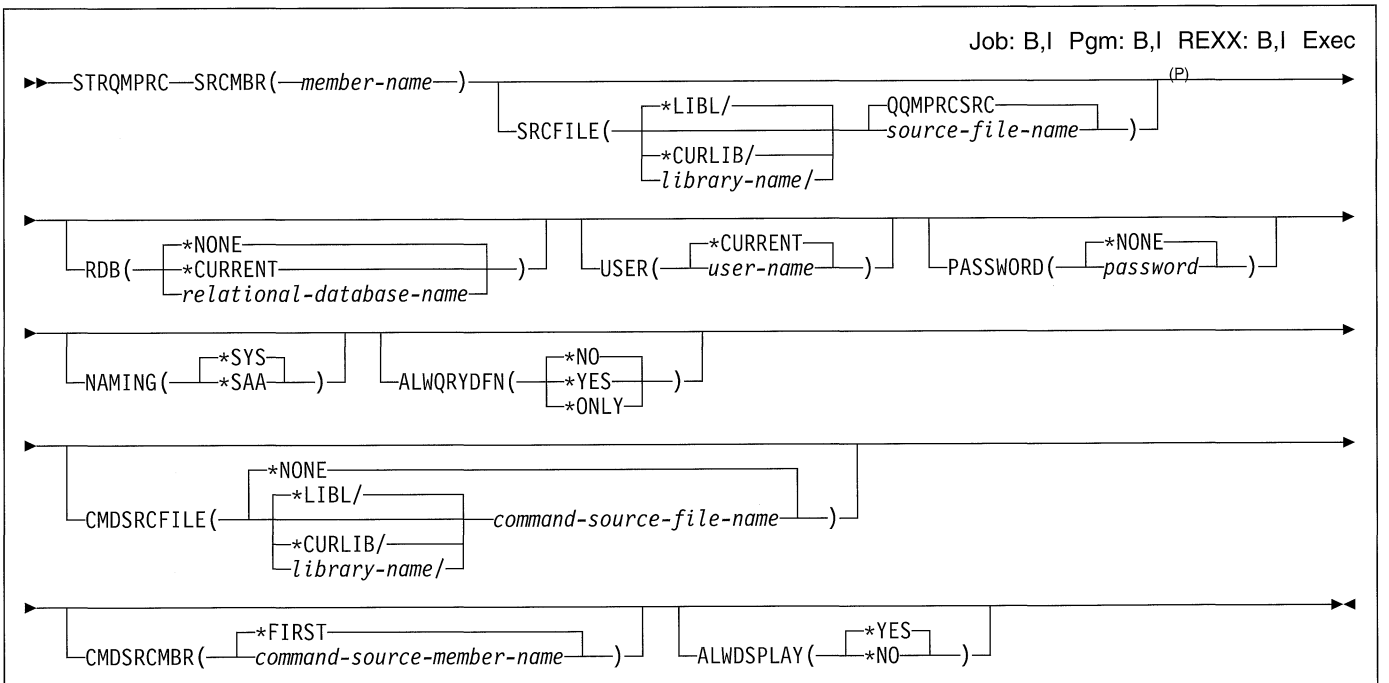
page-number: Specify the number of the page on which printing begins. The page number must be within the page range of the file that is to be printed. This value overrides the value specified on the RESTART parameter on the CHGSPLFA command.

Example

```
STRPRTWTR  DEV(QSYSPRT)  OUTQ(QPRINTS)  WTR(TOM)
```

This command starts a spooling writer named TOM. This writer takes the output from the output queue named QPRINTS and prints the output on the printer named QSYSPRT. Writer messages are sent to the system operator's message queue, and the writer waits for more output when the queue is emptied.

STRQMPRC (Start Query Management Procedure) Command



Note:

(P) All parameters preceding this point can be specified in positional form.

Purpose

The Start Query Management Procedure (STRQMPRC) command allows the user to run a query management procedure that was saved in a source file member.

Required Parameters

SRCMBR

Specifies the name of the member in the source file that contains the query management procedure to be run.

Optional Parameters

SRCFILE

Specifies the qualified name of the source file that contains the query management procedure to run.

The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QQMPRCSRC: The file having the IBM-supplied source file name, QQMPRCSRC, is used.

source-file-name: Specify the name of the source file.

RDB

Specifies the name of the relational database that is accessed during the processing of this command.

***NONE:** The local database is accessed. If the user is connected to a remote database, the connection is reset to local and remains local until completion of this command.

***CURRENT:** The relational database to which the user is currently connected is accessed.

relational-database-name: Specify the name of the relational database that is accessed. The database must have an entry in the relational database directory.

USER

Specifies the user name sent to the remote system when starting the conversation.

***CURRENT:** The user name associated with the current job is used.

user-name: Specify the user name being used for the application requester job.

PASSWORD

Specifies the password to be used on the remote system.

***NONE:** No password is sent. The user name specified on the USER parameter is not valid if this value is specified.

password: Specify the password of the user name specified on the USER parameter.

NAMING

Specifies the naming convention used for naming objects.

***SYS:** The system naming convention is used (library-name/object-name).

***SAA:** The System Application Architecture (SAA) naming convention is used (data-base-name.object-name). If NAMING(*SAA) is specified, CMDSRCFILE(*LIBL) cannot be specified or allowed as a default value for locating any objects specified on other parameters on this command.

ALWQRYDFN

Specifies whether query or form information is taken from a QRYDFN object when no QMQRY or QMFORM object can be found using the specified object name. Any information that has to be derived in this way is discarded when the common programming interface (CPI) command in the procedure is completed. No query management objects are created.

***NO:** The information is not taken from a QRYDFN object.

***YES:** The information is taken from a QRYDFN object when the specified QMQRY or QMFORM object is not found.

***ONLY:** The information is taken only from a QRYDFN object. Query management objects are ignored.

CMDSRCFILE

Specifies the qualified name of the command source file that query management uses to run a command procedure.

***NONE:** A command source file is not used. The CMDSRCMBR parameter is ignored.

The name of the command source file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

command-source-file-name: Specify the name of the command source file.

CMDSRCMBR

Specifies the name of the command source member that query management uses to run a command procedure. A command procedure can only contain query management set commands which set variables that start with the 'DSQ' value.

***FIRST:** The first member is used.

command-source-member-name: Specify the name of the command source member.

ALWDSPLAY

Specifies the display mode used. The query management session is set to interactive mode if ALWDSPLAY(*YES) is specified. If ALWDSPLAY(*NO) is specified, then the session is set to batch mode. The mode is automatically set to batch when running this command in a batch environment.

***YES:** Displays are shown when used in an interactive session. This mode allows the user to interact with the query management commands in the procedure.

***NO:** No displays are shown.

Examples

Example 1: Running a Query Management Procedure

```
STRQMPRC  SRCMBR(MYPROC)  SRCFILE(RPTLIB/PROCFILE)
```

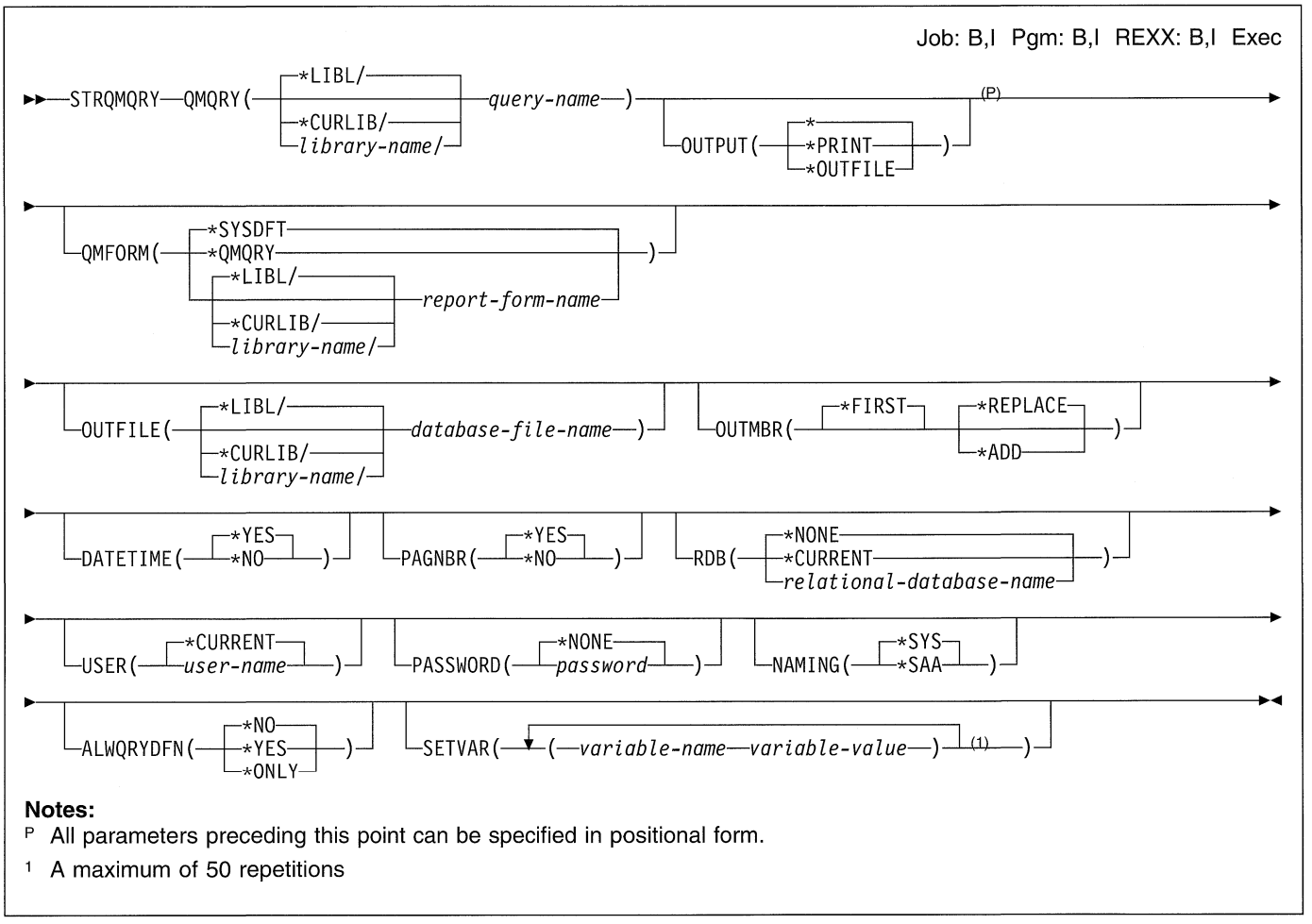
This command starts the query management procedure stored as the member named MYPROC in the source file named PROCFILE in the RPTLIB library.

Example 2: Taking Information From QRYDFN Objects

```
STRQMPRC  SRCMBR(MYPROC)  SRCFILE(PROCFILE)
  ALWQRYDFN(*YES)  ALWDSPLAY(*NO)
```

This command starts the query management procedure stored as the member named MYPROC in the first file named PROCFILE in the library list for the job. Query and form information is allowed to be taken, as needed, from QRYDFN objects when the procedure statements are processed. No reports are shown but they can be printed if the user specifies a print request. Objects are replaced without confirmation if confirmation is not requested by the user. The procedure ends with some errors if processing locates a global variable that is not set or if confirmation was requested before replacing objects that already exist.

STRQMQRY (Start Query Management Query) Command



Notes:

| P All parameters preceding this point can be specified in positional form.

1 A maximum of 50 repetitions

Purpose

The Start Query Management Query (STRQMQRY) command is used to run a query.

The user of this command must first identify the query that is to be processed. The query is any single Structured Query Language (SQL) statement in a QMQRY object. The SQL statement can also be taken from a query definition (QRYDFN) object when a QMQRY object does not exist.

The user can show the output on the display, print it, or store it in a database file.

If the SQL statement inside the query does not create an answer-set, then no report or output file is created. This happens if the SQL statement inside a query is not valid or the SQL statement is not a SELECT clause.

If the query contains substitution variables, the SETVAR parameter can be used to set the variables for the query. If prompting is enabled, query management asks the user to provide a value for each variable that was not set.

Required Parameters

QMQRY

Specifies the name of the query management query to be run. This parameter is required when running a query.

| The name of the query can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

query-name: Specify the name of the query to be run.

Optional Parameters

OUTPUT

Specifies whether the output from the command is shown at the requesting work station, printed with the job's spooled output, or directed to a database file.

***:** The output produced by the query is formatted with the specified report form and, in interactive mode, sent to the work station that runs the command. If the command is run in batch mode, the output is sent to the default printer used by query management.

***PRINT:** The output produced by the query is formatted with the specified query management form, then sent to the default printer used by query management.

***OUTFILE:** The output produced by the query is written to a database file (table), which is inserted into a collection.

QMFORM

Specifies which query management report form is to be applied to the answer-set to format the printed or displayed output.

***SYSDFT:** A default report form is created and used for the report that is printed or displayed.

***QMQRY:** The value specified on the QMQRY parameter is used to locate the report form.

The name of the report form can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

report-form-name: Specify the name of the report form.

OUTFILE

Specifies the database file that receives the query output. If the file specified does not exist, the system creates it in the specified library as a table in a collection. If the file is created by this function, the authority for users without specific authority is *EXCLUDE.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file that receives the output of the command.

OUTMBR

Specifies the name of the database file member to which the output is directed.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output.

Element 2: Operation to Perform on Member

***REPLACE:** The file is cleared before new records are inserted.

***ADD:** New records are added after any existing records.

DATETIME

Specifies whether the system date and time are printed on the bottom of each page.

***YES:** The system date and time are printed on the bottom of each page.

***NO:** The system date and time are not printed on the bottom of each page.

PAGNBR

Specifies whether page numbers are printed on the bottom of each page.

***YES:** The page numbers are printed on the bottom of each page.

***NO:** The page numbers are not printed.

RDB

Specifies the name of the relational database that is accessed during the processing of this command.

***NONE:** The local database is accessed. If the user is connected to a remote database, the connection is reset to local and remains local until completion of this command.

***CURRENT:** The relational database to which the user is currently connected is accessed. If the user is connected to a remote database, *OUTFILE cannot be specified on the OUTPUT parameter.

relational-database-name: Specify the name of the relational database that is accessed. The database must have an entry in the relational database directory.

If OUTPUT(*OUTFILE) is specified, the database file to which the output is directed is accessed on the local database. If the relational database specified is a remote database and OUTPUT(*OUTFILE) is specified, the connection is reset to local for the *OUTFILE processing and remains local upon completion of the STRQMQRY command.

USER

Specifies the user name sent to the remote system when starting the conversation.

***CURRENT:** The user name associated with the current job is used.

user-name: Specify the user name being used for the application requester job.

PASSWORD

Specifies the password to be used on the remote system.

***NONE:** No password is sent. The user name specified on the USER parameter is not valid if this value is specified.

password: Specify the password of the user name specified on the USER parameter.

NAMING

Specifies the naming convention used for naming objects.

***SYS:** The system naming convention is used.

***SAA:** The System Application Architecture (SAA) naming convention is used. If NAMING(*SAA) is specified, the *LIBL value cannot be specified or allowed to be a default value for locating any objects specified on other parameters on this command.

ALWQRYDFN

Specifies whether query or form information is taken from a QRYDFN object when a QMQRY or QMFORM object cannot be found using the object name specified by the user. Any information that has to be derived in this way is discarded when the command has completed processing. No query management object is created.

***NO:** Information is not taken from a QRYDFN object.

***YES:** Information is taken from a QRYDFN object when the specified QMQRY or QMFORM object name is not found.

***ONLY:** Information is taken only from a QRYDFN object. Query management objects are ignored.

SETVAR

Specifies the variables that are set by query management before the query is run. Up to 50 variables can be set.

Element 1: Variable Name

variable-name: Specify a variable name. Valid values range from 1 to 18 characters. Since lower case characters in variable names are changed to upper case when passed to the command processing program, the user cannot use this parameter to set values for variables with mixed case names.

Element 2: Variable Value

variable-value: Specify a variable value. Valid values range from 0 to 55 characters. If the user encloses a value in apostrophes, the apostrophes are removed and double apostrophes within the value are condensed to single apostrophes when the value is passed to the command processing program.

Examples

Example 1: Displaying Query Output

```
STRQMQRY  QMQRY(MYLIB/MYQRY)  QMFORM(FORM1)
```

This command runs query management query MYQRY located in library MYLIB. The library list is searched for form FORM1, which is used for the output sent to the display.

Example 2: Taking Information From Either QMQRY or QRYDFN

```
STRQMQRY  QMQRY(MYLIB/MYQRY)  QMFORM(FORM1)  ALWQRYDFN(*YES)
```

This command runs query management query (QMQRY) MYQRY located in library MYLIB. If QMQRY object MYQRY is not found in library MYLIB, the information is taken from query definition (QRYDFN) MYQRY located in library MYLIB. The library list is searched for query management form FORM1 whose information is used to format the output. If QMFORM object FORM1 is not found in the library list, the library list is searched for QRYDFN FORM1, and that information is used to format the output shown on the display.

Example 3: Printing Query Output

```
STRQMQRY  QMQRY(MYLIB/QUERY1)  OUTTYPE(*PRINTER)
```

This command runs query QUERY1 located in library MYLIB. The report is formatted and printed on the printer specified in the printer file associated with the query session.

Example 4: Sending Output to an Existing File

```
STRQMQRY  QMQRY(*CURLIB/MYQRY)  OUTPUT(*OUTFILE)
  OUTFILE(MYTAB)  OUTMBR(*FIRST *ADD)
```

This command runs the query named MYQRY located in the current library for the user's job. The selected data records are added to the previously created table named MYTAB in collection MYCOL.

Example 5: Running a Query Containing Substitution Variables

```
STRQMQRY  QMQRY(MYQUERY)
  SETVAR((VAR1 'select * from mytable')
  (VAR2 'where salary > 15000'))
```

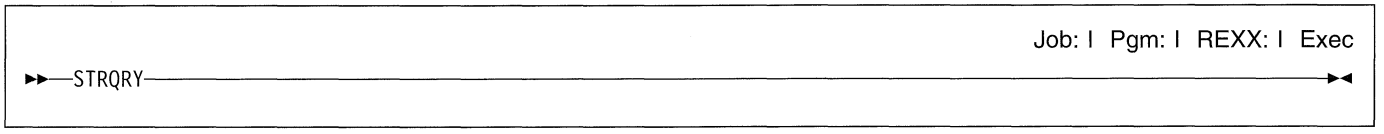
This command runs query MYQUERY, which contains only substitution variables, &VAR1 and &VAR2. These two variables contain the entire structured query language (SQL) statement.

Example 6: Changing a Variable

```
STRQMQRY  QMQRY(QRYNAME)  SETVAR((LASTNAME '''Smith'''))
```

This command runs query QRYNAME, setting the variable LASTNAME to the value, 'Smith'.

STRQRY (Start Query) Command



Purpose

The Start Query (STRQRY) command shows the Query/400 main menu.

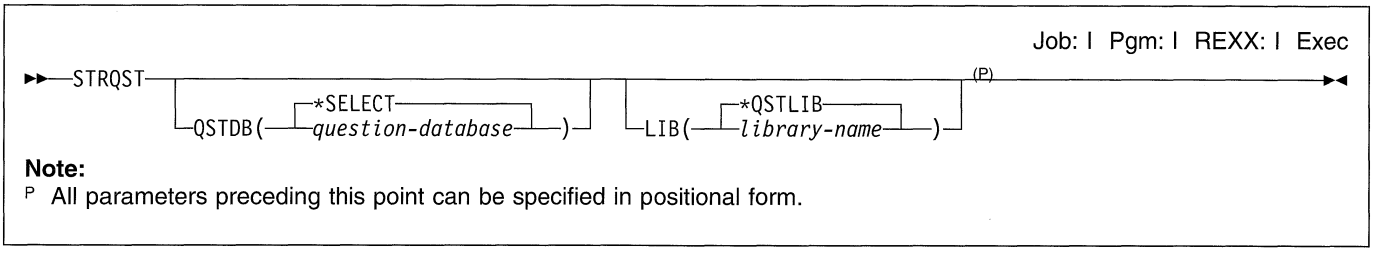
There are no parameters for this command.

Example

```
STRQRY
```

This command shows the main Query menu.

STRQST (Start Question and Answer) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Start Question and Answer (STRQST) command shows the user the main Question & Answer (Q & A) menu. More information is in the *Q & A Database Coordinator's Guide*

Optional Parameters

QSTDB

Specifies the Q & A database with which to work.

***SELECT:** The user is asked to specify a Q & A database. If only one Q & A database exists on the system, it is the default.

question-database: Specify the name of the Q & A database with which to work.

LIB

Specifies the name of the library that contains the Q & A database.

***QSTLIB:** The library containing the specified Q & A database is searched. If *SELECT is specified on the QSTDB parameter, any Q & A database in any library for which the user is authorized can be selected.

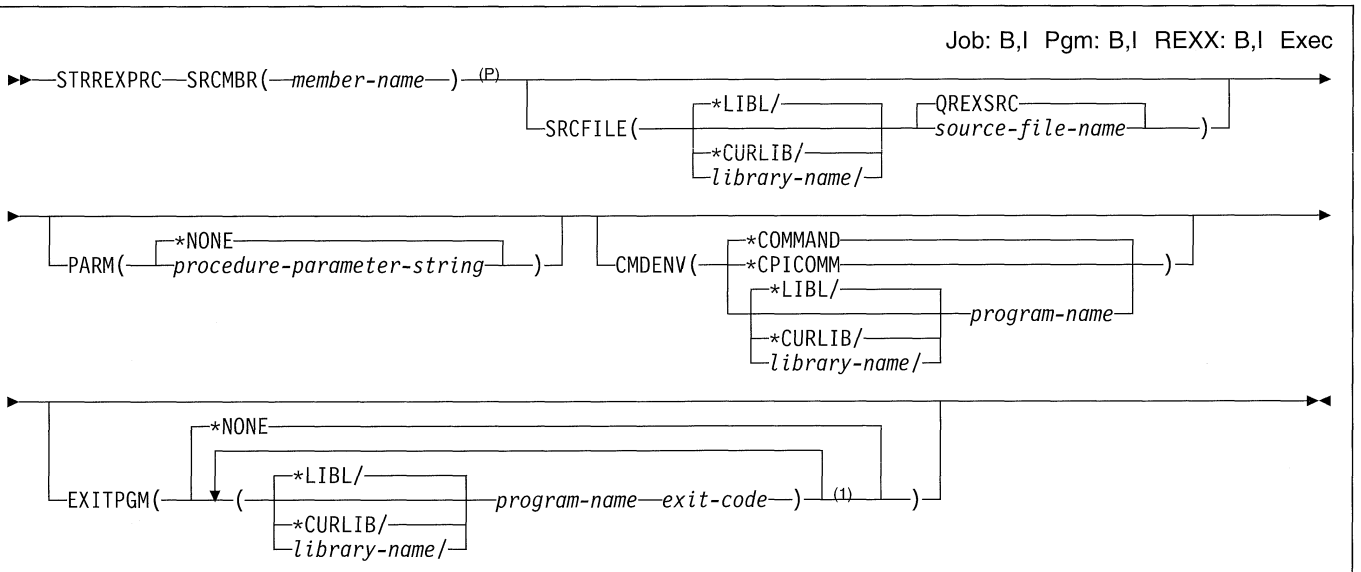
library-name: Specify the name of the library to be searched. If *SELECT is specified on the QSTDB parameter, any database in the library for which the user is authorized can be selected.

Example

```
STRQST
```

This command shows the Question and Answer (Q & A) main menu.

STRREXPRC (Start REXX Procedure) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 A maximum of 8 repetitions

Purpose

The Start REXX Procedure (STRREXPRC) command calls the REXX interpreter to start a REXX procedure that resides in the specified source member.

Required Parameters

SRCMBR

Specifies the name of the source file member that contains the REXX procedure to be started.

Optional Parameters

SRCFILE

Specifies the source file containing the REXX procedure to be started.

The name of the source file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QREXSRC: The IBM-supplied source file QREXSRC contains the REXX procedure to be started.

source-file-name: Specify the name of the source file that contains the REXX procedure to be started.

PARM

Specifies procedure parameter values passed to the REXX procedure when it is started. These values are accessed through the argument (ARG) instruction within the REXX procedure.

***NONE:** There are no procedure parameters for the REXX procedure. The ARG instruction returns a null string.

procedure-parameter-string: Specify the parameter string to be passed to the REXX procedure. Up to 3000 characters are allowed.

CMDENV

Specifies the initial command environment program name to process commands that are merged within the REXX procedure. The REXX interpreter calls this environment whenever a command is encountered during the REXX procedure.

***COMMAND:** The AS/400 system command environment is used.

***CPICOMM:** The Common Programming Interface (CPI) for Communications command environment is used.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of the program.

EXITPGM

Specifies exit programs to be used when the interpreter is called. A maximum of 8 exit programs can be specified.

***NONE:** There are no exit programs for the interpreter being called.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

Element 1: Exit Program

program-name: Specify up to eight program names.

Element 2: Exit Code

exit-code: Specify up to eight exit codes. The valid exit codes that can be specified for this value range from 2 through 5 and 6 through 8.

Exit-code Description

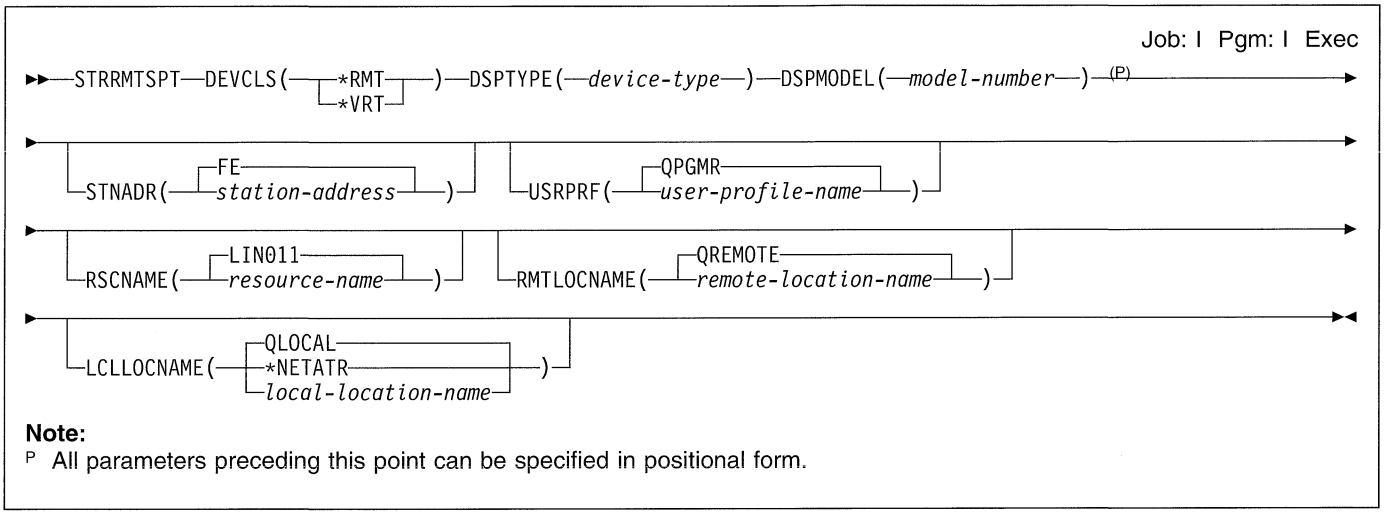
1. The associated program is called whenever an external function or subroutine is called by the REXX program. The exit program is then responsible for locating and calling the requested routine.
2. The associated program is called whenever the interpreter is going to call a command. The exit program is responsible for locating and calling the command given the command string and the current environment name.
3. The associated program is called whenever a REXX instruction or function attempts an operation on the REXX external data queue.
4. The associated program is called when session input or output operations are attempted.
5. The associated program is called after running each clause of the REXX procedure to determine whether it is to be halted.
6. The associated program is called after running each clause of the REXX program to check if tracing is to be turned on or off.
7. The associated program is called before interpretation of the first instruction of a REXX procedure (including REXX procedures called as external functions and subroutines).
8. The associated program is called after interpretations of the last instruction of a REXX procedure (including REXX procedures called as external functions and subroutines).

Example

```
STRREXPRC  SRCMBR(ABC)
```

This command calls the REXX interpreter instructing it to run the source member named ABC in the first QREXSRC source file in the library list (*LIBL).

STRRMTSPT (Start Remote Support) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Start Remote Support (STRRMTSPT) command creates and varies on all configuration objects needed for remote support. Remote support allows the service organization to access your system from a remote work station. Remote support options for pass-through and remote work station operations are provided. If any existing remote support configuration objects are found, they are deleted and then re-created. After the configuration objects have been created, they are varied on. You must provide a telephone number, user ID, and password before the support person can sign on your system.

Restriction: The remote work station used by the service organization must be one of those listed in the DSPTYPE and DSPMODEL parameters. If your support organization has a configuration that does not match that of the remote work station, you must work with the support person to create the correct configuration of objects on your system.

Required Parameters

DEVCLS

Specifies the device class for this display station.

***RMT:** This device description is for a device connected to a remote work station. The configuration objects created for this option include a line, controller, display, and printer description.

***VRT:** This device description is for a virtual device. The configuration objects created for this option include a line, controller, display, virtual controller, and virtual display description.

DSPTYPE

Specifies the work station device type which is used for remote support. The possible device types are:

3179 Display Station
3180 Display Station
3196 Display Station
3197 Display Station
5251 Display Station
5291 Display Station

DSPMODEL

Specifies the model number of the device for this description. The possible model numbers for each device type are:

Device Type	Model
3179	2
3180	2
3196	A1, A2, B1, B2
3197	C1, C2, D1, D2, W1, W2
5251	11
5291	1, 2

Optional Parameters

STNADR

Specifies the hexadecimal address by which the local system is known to the remote system. The hexadecimal address is the polling address assigned to this system.

FE: The hexadecimal value FE is the local system address.

station-address: Specify a hexadecimal value ranging from 01 through FE.

USRPRF

Specifies the user profile that the IBM service personnel will use to sign on the customer's system. This profile is made the owner of the line, controller, and device descriptions created by the system when remote support is started. The user profile must already exist on the customer's system.

QPGMR: The system-supplied user profile QPGMR is used to sign on the customer's system.

user-profile-name: Specify the name of the user profile that will be used to sign on the remote system.

RSCNAME
Specifies the name of the resource that the customer actually uses to access the AS/400 electronic customer support.

LIN011: The shipped default for accessing electronic customer support is used.

resource-name: Specify the name of the resource used to access electronic customer support.

RMTLOCNAME
Specifies the remote location name of the system with which this object communicates.

QREMOTE: The system-supplied remote location name is used.

remote-location-name: Specify the full name of a remote location.

LCLLOCNAME
Specifies the local location name.

Note: The name cannot be the same as that specified for the RMTLOCNAME parameter. The combination of the local and remote location names must be unique for each device attached to the same controller.

QLOCAL: The system-supplied local location name is used.

***NETATR:** The LCLLOCNAME value specified in the system network attributes is used.

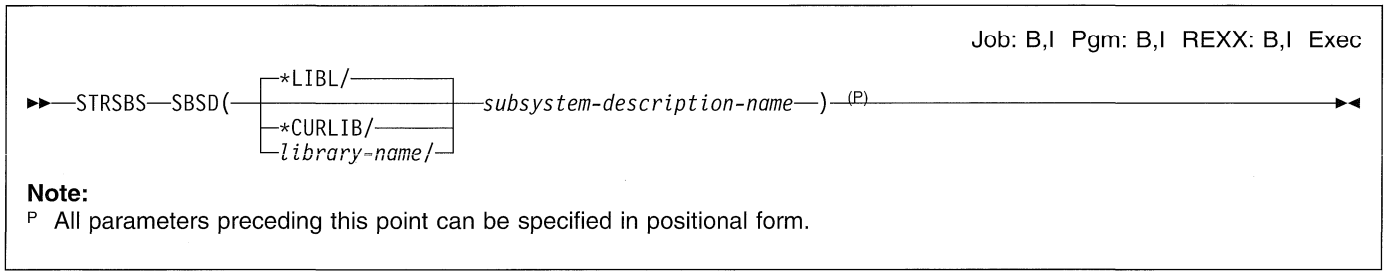
local-location-name: Specify the name of the local location.

Example

```
STRRMTSPT  DEVCLS  DSPTYPE(3180)  DSPMODEL(2)
  STNADR(A1)
```

This command creates and varies on the 3180 Model 2 Display Station located at station address ADR01.

STRSBS (Start Subsystem) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Start Subsystem (STRSBS) command starts a subsystem based on the subsystem description specified in the command. Once started, the subsystem is known by the unqualified name of the subsystem description (the subsystem description name without the library name). When the subsystem is started, the system allocates the necessary and available resources (storage, work stations, and job queue) that are specified in the subsystem description.

Storage is allocated to the subsystem according to the storage pool definitions specified in the subsystem description, starting with the lower numbered storage pool definitions. If all the pool definitions cannot be allocated, because the maximum number of storage pools on the system is reached or because insufficient storage is available, messages indicating which pools could not be allocated are sent to the system operator. If storage becomes available later, or if the number of active storage pools is reduced, the available resources are automatically allocated to the subsystem to satisfy its unfulfilled requirements. Any jobs that would normally run in a storage pool that is not allocated are run in the shared storage pool (*BASE).

Allocating Work Stations: Work stations are allocated to the subsystem according to the work station entries in the subsystem description. Each work station whose name (or type, if not specified by name) is contained in one of the subsystem description's work station entries, and whose entry specifies AT(*SIGNON), is allocated to this subsystem unless it is currently signed on to another subsystem. The sign-on prompt is displayed on each work station that is allocated. Work stations that are already signed on in another subsystem remain allocated to that subsystem until the subsystem that allowed the sign-on is ended, or until the user transfers the job to this subsystem. (Messages indicating the names of the work stations that could not be allocated are sent to the system operator.)

If multiple subsystems specify the same work station in their work station entries, each subsystem, as it is started, attempts to allocate that work station. Each successive subsystem allocates that work station unless a user signs on while the work station is allocated to one of the previously started subsystems. When a signed-on work station is signed off, it still remains allocated to the same subsystem until another subsystem is started that specifies that work station. If, however, a work station is varied offline and several active subsystems specify that work station, the subsystem to which the work station is allocated when it is varied online is unpredictable.

Allocating Job Queues: If a job queue is specified in the work entries of the subsystem description, the job queue is allocated to the subsystem. If the job queue does not exist or if it is already allocated to an active subsystem, no job queue is allocated to the subsystem and a message is sent to the system operator. If the job queue later becomes available, it is *not* automatically allocated to the subsystem. To allocate the job queue to the subsystem, the subsystem must end, then start again.

Restriction: To start a subsystem, the user must have object operational authority to use the subsystem description.

Required Parameters

SBSD

Specifies the qualified name of the subsystem description that defines the operational environment (subsystem) being started.

Note: The IBM-supplied object named QLPINSTALL is not valid on this parameter.

The name of the subsystem description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

subsystem-description-name: Specify the name of the subsystem description that defines the subsystem being started.

The name of the subsystem description cannot be the same as the name of a subsystem that is currently active, even though the subsystem descriptions are in different libraries.

Examples

Example 1: Starting a Subsystem

```
STRSBS  SBSD(QBATCH)
```

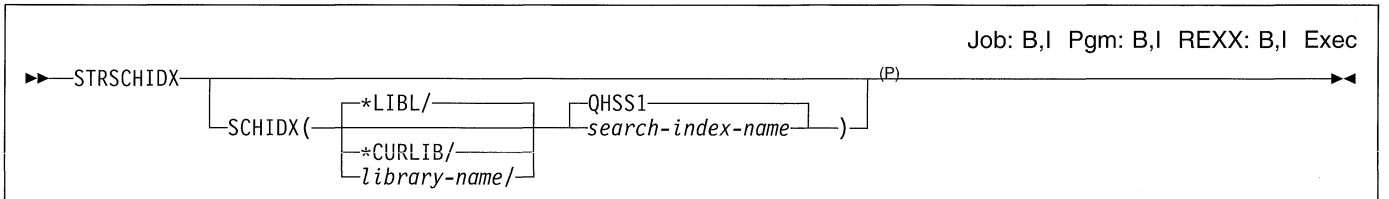
This command starts the batch subsystem named QBATCH.

Example 2: Starting a Subsystem

```
STRSBS  SBSD(QGPL/TELLER)
```

This command starts the subsystem that is associated with the TELLER subsystem description in the QGPL library. The subsystem name is TELLER.

STRSCHIDX (Start Search Index) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Start Search Index (STRSCHIDX) command is used to access a search index without using the Help key or the F11 key.

Restrictions: The user of this command must have *USE authority for the search index.

Optional Parameters

SCHIDX

Specifies the qualified name of the search index from which the index entries are displayed. If this parameter is not used, the main help index for the AS/400 system is used.

The name of the search index can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QHSS1: QHSS1, the main help index for the AS/400 system, is used.

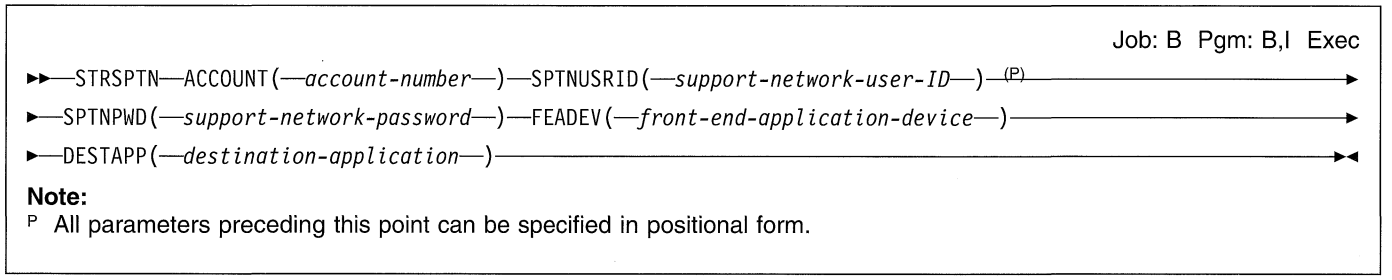
search-index-name: Specify the name of the search index.

Example

```
STRSCHIDX
```

This command accesses the main help index for the AS/400 system.

STRSPTN (Start Support Network) Command



Purpose

The Start Support Network (STRSPTN) command opens a communications path through the remote support network to the specified destination application.

This command is provided for customers who want to write their own programs to interface with one of the remote support systems.

Required Parameters

ACCOUNT
Specifies the user's support network organization account number. The organization account ID must be registered with the support network.

SPTNUSRID
Specifies the network user ID in the account specified in the ACCOUNT parameter. The user ID must be registered with the support network and must be valid for the specified account.

SPTNPWD
Specifies the network password for the specified user. The password must be registered with the support network and must be valid for the specified user.

Note: A support network password can expire. The user must then change the password interactively by using the Work with Product Information (WRKPRDINF) command.

FEADEV
Specifies the description of the device used as the Front End Application (FEA).

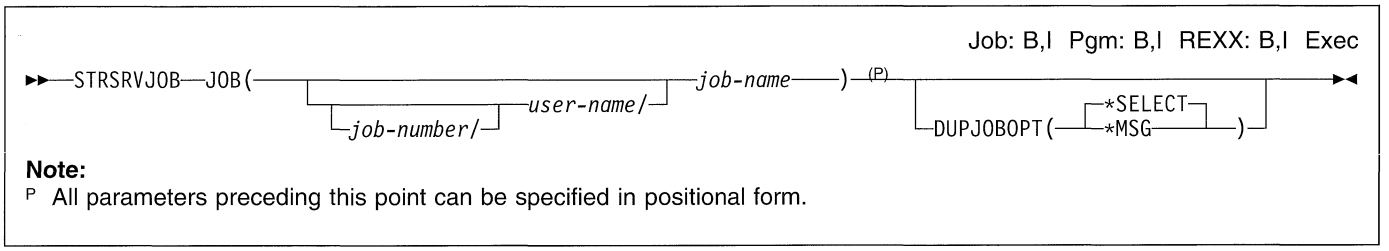
DESTAPP
Specifies the destination application (DESTA) to which the communication path is connected.

Example

```
STRSPTN  ACCOUNT(11420880)  SPTNUSRID(ACME)
  SPTNPWD(11111)  FEADEV(QTIFEA)  DESTAPP(AAAAAA)
```

This command establishes a communication path through the remote support network, for user ID ACME operating under password 11111 at account 11420880. The path allows access to application AAAAAA.

STRSRVJOB (Start Service Job) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Start Service Job (STRSRVJOB) command starts the remote service operation for a specified job (other than the job issuing the command) so that other service commands can be entered to service the specified job. Dump, debug, and trace commands can be run in that job until service operation ends. Service operation continues until the End Service Job (ENDSRVJOB) command is run.

The following commands can be used in the serviced job while remote service is being done:

Dump Job (DMPJOB)
Dump Job Internal (DMPJOBINT)
Dump Object (DMPOBJ)
Dump System Object (DMPSYSOBJ)
Start Debug (STRDBG)
Trace Internal (TRCINT)
Trace Job (TRCJOB)
Trace ICF (TRCICF)

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Required Parameters

JOB

Specifies the name of the job being serviced. If no job number is given, all of the jobs currently in the system are searched for the simple job name. If duplicates of the specified name are found, messages are sent to the user, and the user name and job number must be specified. The job name entered cannot be the name of the job issuing the command.

A job identifier is a qualified name with up to three elements. For example:

```
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

job-name: Specify the name of the job being serviced.

user-name: Specify the name of the user of the job being serviced.

job-number: Specify the number of the job being serviced.

DUPJOBOPT

Specifies the action taken when duplicate jobs are found by this command.

***SELECT:** The selection display is shown when duplicate jobs are found during an interactive session. Otherwise, an escape message is issued.

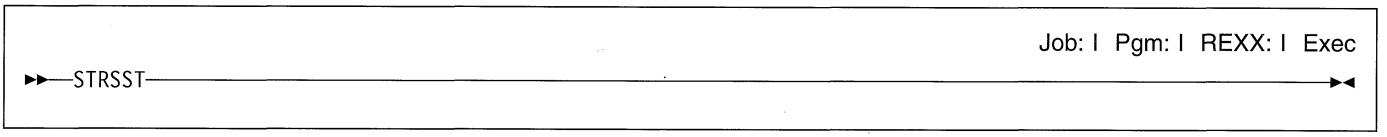
***MSG:** An escape message is issued when duplicate jobs are found.

Example

```
STRSRVJOB  JOB(ABCD)
```

This command starts the remote service operation so that any trace, debug, or dump commands entered in this job are applied to the job named ABCD.

STRSST (Start System Service Tools) Command



Purpose

The Start System Service Tools (STRSST) command shows the System Service Tools (SST) menu.

The user can:

- Start a service function
- Work with disk unit configuration and data
- Work with active service functions
- Work with diskette data recovery

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. The QSRV user profile has private authorities to use the command.
3. You must have *SERVICE special authority to use this command.

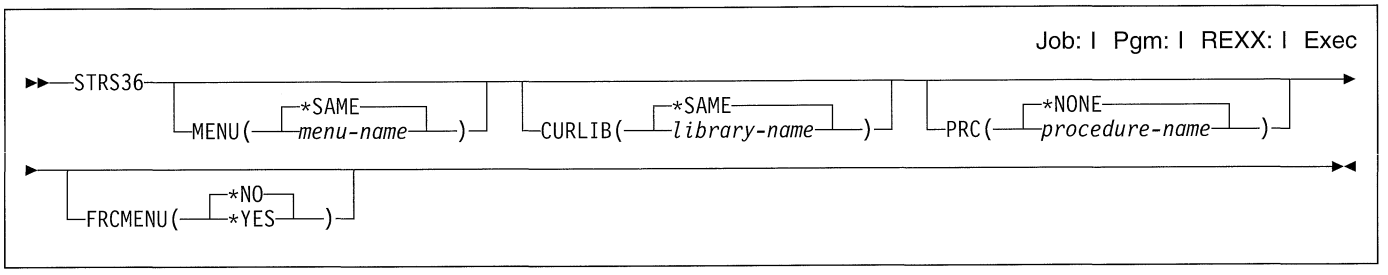
There are no parameters for this command.

Example

```
STRSST
```

This command shows the Start System Service Tools menu.

STRS36 (Start System/36) Command



Purpose

The Start System/36 (STRS36) command starts a System/36 environment session (if one is not already active). Even if the System/36 environment is active, this command allows the user to show a menu or run a program or procedure before showing a menu.

When this command ends, the System/36 environment returns to the active or inactive state from which this command is run.

- If the System/36 environment is active when this command is issued, the environment remains active when the command ends. You can press the Exit or Previous key at the menu to return to the point at which the command was issued.
- If the System/36 environment is inactive when this command is issued, the environment again becomes inactive when the command ends. You must use the End System/36 (ENDS36) command to exit from the menu.

Restrictions: This command cannot be used if a System/36 procedure is already in process. This command cannot be placed in a procedure or in a program that is started by a procedure.

Optional Parameters

MENU

Specifies the first menu that is shown when the System/36 environment is started.

***SAME:** The menu specified in the job does not change. If no menu is specified in the job, the initial menu specified in the user profile is shown.

menu-name: Specify the name of the menu that is shown first when the System/36 environment is started.

CURLIB

Specifies the name of the library to use for the current library in the System/36 environment.

***SAME:** The current library does not change. If the current library is *CRTDFT, the current library is set to #LIBRARY.

library-name: Specify the name of the library to use for the current library in the System/36 environment.

PRC

Specifies the name of the procedure or program to run before the menu is shown.

***NONE:** No procedure or program runs.

procedure-name: Specify the name of the procedure or program to run.

FRCMENU

Specifies whether a menu is shown and what this command does in an active System/36 environment.

***NO:** The specified menu is not shown if the System/36 environment is active when this command is run. This command does nothing, and the user is returned to the point at which the command was issued.

***YES:** The specified menu is shown even if the System/36 environment is active when this command is run. The current library is set and the program or procedure is run as specified in this command.

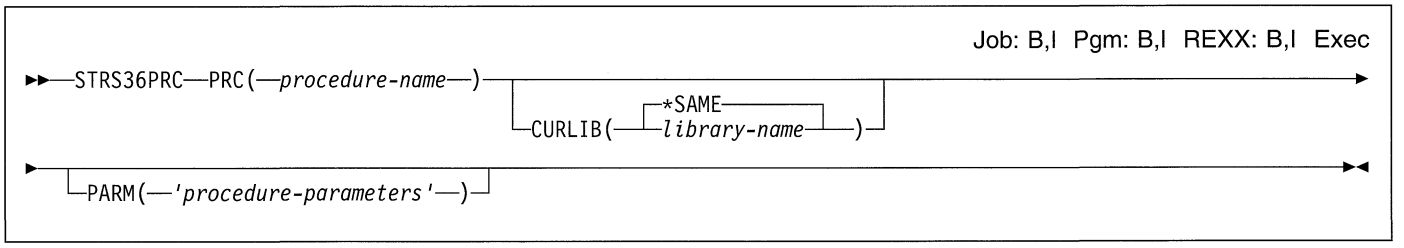
Example

```
STRS36  MENU(user)  CURLIB(MYLIB)  PRC(INITPROC)
```

This command starts a System/36 environment session. This command:

- Displays the user menu
- Changes the current library to MYLIB
- Runs procedure INITPROC before showing the user menu

STRS36PRC (Start System/36 Procedure) Command



Purpose

The Start System/36 Procedure (STRS36PRC) command starts a System/36 procedure. It is valid whether or not the System/36 Environment is active, but it is not valid if a System/36 procedure is already running. It cannot be placed in a procedure or in a program that is called by a procedure.

Required Parameters

PRC

Specifies the name of the System/36 procedure to run.

Optional Parameters

CURLIB

Specifies the name of the library being used as the current library during the processing of this command.

***SAME:** The value does not change.

library-name: Specify the name of the library to use for the current library while running the System/36 procedure.

PARM

Specifies procedure parameters for the procedure. Procedure parameters allow information to be passed to the procedure. If no parameters are specified, no parameters are passed to the procedure.

Examples

Example 1: Changing the Current Library

```
STRS36PRC  PRC(PROC1)  CURLIB(MYLIB)
```

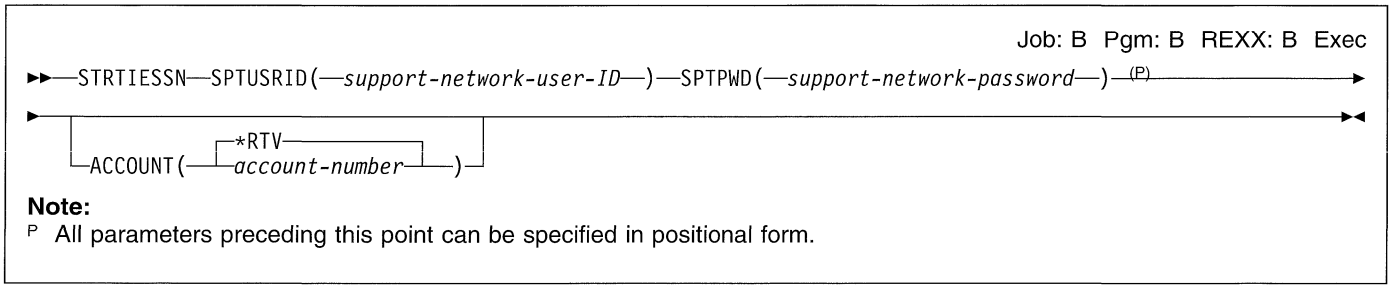
This command changes the current library to MYLIB and runs procedure PROC1.

Example 2: Listing Files Used by the System

```
STRS36PRC  PRC(CATALOG)  PARM('ALL,F1')
```

This command lists all files used by the System/36 Environment.

STRTIESSN (Start Technical Information Exchange Session) Command



Note:
(P) All parameters preceding this point can be specified in positional form.

Purpose

The Start Technical Information Exchange Session (STRTIESSN) command establishes the communications link for a TIE batch session. This command must precede other TIE batch commands.

Required Parameters

SPTUSRID
Specifies the user ID used to sign on the remote support network.

SPTPWD
Specifies the password used to sign on the remote support network.

Optional Parameter

ACCOUNT
Specifies the network account number used to sign on the remote support network. If the account number is not specified, the account number from the contact database is used.

***RTV:** The account number from the contact database is used.

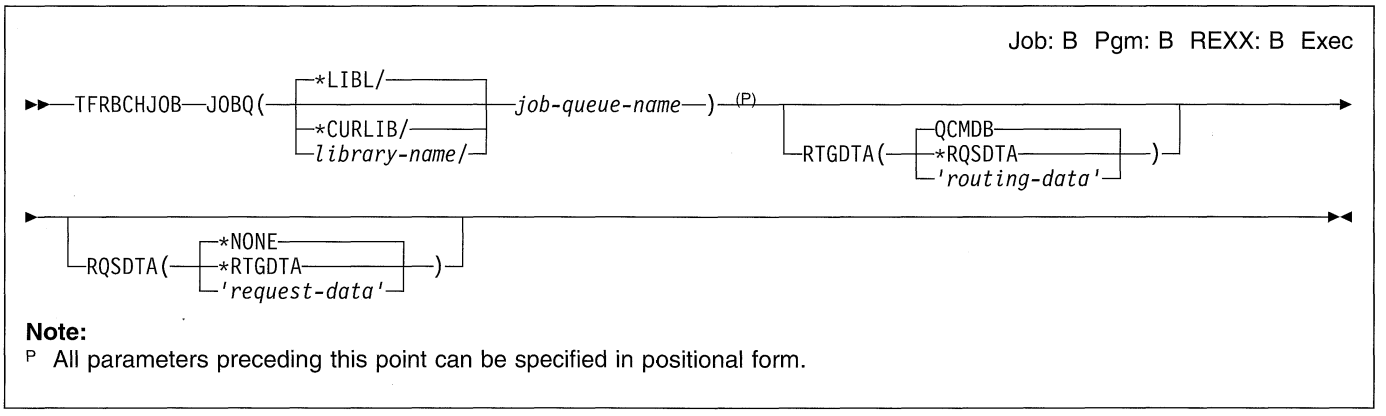
account-number: Specify the account number that is used to sign on the remote support network.

Example

```
STRTIESSN  SPTUSRID(ACME)  SPTPWD(11111)
  ACCOUNT(11420880)
```

This command displays the TIE main menu for account number 11420880.

TFRBCHJOB (Transfer Batch Job) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Transfer Batch Job (TFRBCHJOB) command transfers a batch job to the specified job queue. The job queue does not have to be allocated to an active subsystem at the time of the batch job transfer. The batch job that is transferred is the one in which this TFRBCHJOB command is issued. Routing data and request data can be specified for the batch job when it is transferred. The routing data specified is processed in the subsystem in which the job queue is active. The request data is placed after all other request data for the job. The transferred batch job resumes running the request data immediately following the TFRBCHJOB command.

If the TFRBCHJOB command is issued in a CL program, all subsequent commands in the CL program are bypassed. If objects that are allocated to the current routing step or files that are open in the current routing step are needed in the new routing step, they must be allocated or opened again after the new routing step has been started.

Note: If you are working in a System/36 environment, the TFRBCHJOB command does not transfer the System/36 environment to the new job.

A batch job transferred to a job queue by the TFRBCHJOB command exists through an initial program load (IPL) if the batch job was residing on the job queue at the time the system was powered down; the job's temporary objects exist through a power down. A batch job's temporary objects are destroyed during the power down if the job was transferred by the Transfer Job (TFRJOB) command.

The QTEMP library of a batch job that has been transferred by the TFRBCHJOB command is always empty when the next routing step is started. Caution must be used with the library list in conjunction with a batch job that was transferred to a job queue by the TFRBCHJOB command. The TFRBCHJOB function saves the library list to recover the job on a job queue if an IPL occurs. When the routing step for the transferred batch job is started, the libraries in the saved library list must exist in the system or the job's routing step ends.

Restrictions: (1) The user must have add and read authority for the job queue and operational authority for the subsystem to which the job queue is allocated. (2) The job being transferred must be a batch job that started from a job queue. (3) The TFRBCHJOB command is not allowed to run in a batch communications job (a batch job that was started as the result of a program start request) or a batch immediate job. Also, interactive, auto-start, reader, writer, and system jobs cannot be transferred by the TFRBCHJOB command.

Required Parameter

JOBQ

Specifies the qualified name of the job queue to which the batch job is transferred.

The name of the queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-queue-name: Specify the name of the job queue to which the batch job is transferred.

Optional Parameters

RTGDTA

Specifies the routing data used with this job description to start jobs. The routing data is used to determine the routing entry (in the subsystem description) that identifies the program where the job runs.

QCMDB: This routing data matches a routing entry in the IBM-supplied subsystem description, which starts a routing step processed by the IBM-supplied control language processor QCMD.

***RQSDTA:** Up to 80 characters of the request data specified in the RQSDTA parameter of this command are used as the routing data for the routing step. If this value is used, RQSDTA(*RTGDTA or *NONE) cannot be specified.

'routing-data': Specify the character string that is used as the routing data for starting the routing step. Up to 80 characters can be specified, enclosed in apostrophes if blanks or special characters are included.

RQSDTA

Specifies the request data that is added to the end of the job's message queue for use by the new routing step.

***NONE:** No request data is placed in the job's message queue.

***RTGDTA:** The routing data specified in the RTGDTA parameter is placed at the end of the job's message queue. If this value is used, RTGDTA(*RQSDTA) cannot be specified.

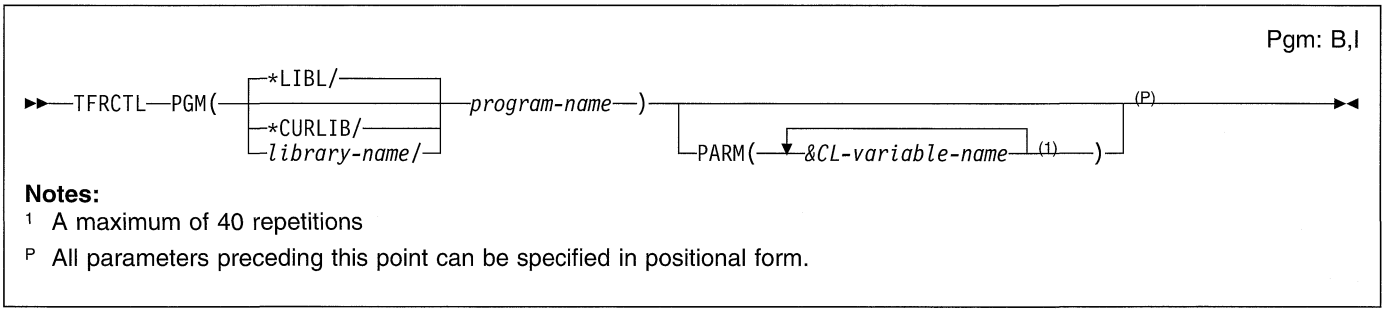
'request-data': Specify the character string that is placed at the end of the job's message queue for use by the new routing step or some subsequent routing step in the job. Up to 256 characters can be specified, enclosed in apostrophes if blanks or special characters are included. When a CL command is specified, it must be enclosed in single apostrophes, and where apostrophes would normally be used in the command, double apostrophes must be used.

Example

```
TFRBCHJOB  JOBQ(QGPL/QCTL)  RTGDTA(APPLICS)
```

This command transfers the batch job where the command is entered to the QCTL job queue that is in the QGPL library. The job is routed using the routing data APPLICS. The job must be a batch job.

TFRCTL (Transfer Control) Command

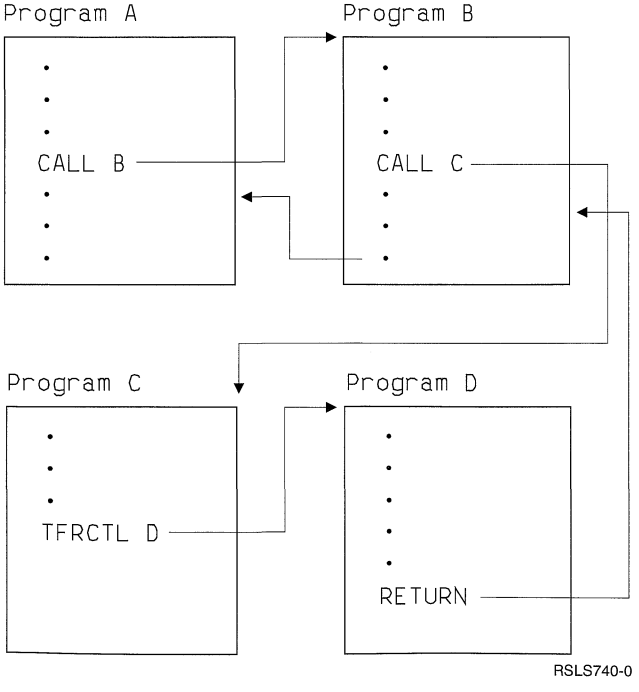


Notes:
1 A maximum of 40 repetitions
P All parameters preceding this point can be specified in positional form.

Purpose

The Transfer Control (TFRCTL) command calls the program specified on the command, passes control to it, and removes the transferring program from the return stack. Because the transferring program is removed from the call stack, control does not return to it when the called program returns control. Instead, control is returned to the command following the last call to the transferring program.

The following diagram shows the operation of both the CALL and TFRCTL commands. First, Program A calls Program B. Program B then calls Program C, which in turn transfers control (using the TFRCTL command) to Program D. In transferring control to Program D, Program C is removed from the call stack; therefore, Program D returns to Program B (to the command following the most recent call to the transferring program, Program C). Finally, Program B returns to Program A.



If the transferring program was created with USRPRF(*OWNER), the authority does not transfer. However, if a program created with USRPRF(*OWNER) calls a program that transfers control, the authority *does* transfer because the program that called the transferring program remains in the call stack.

Optionally, the transferring program passes parameters to the program being called. The storage space used by the CL variables in the transferring program is freed and is available for use by the program receiving control.

The parameter values must be passed in CL variables. Values cannot be passed as constants, as null parameters (that is, parameters whose values are null, specified by *N), as lists of values, or as CL variables that were not specified as parameters on the PGM command that identified the start of the transferring program.

The transferring program can only pass CL variables that were previously passed to it. No CL variable can be passed that exists within the transferring program itself. Up to 40 parameters can be passed to the called program. The parameters passed must agree in type, length, number, and order with those expected by the receiving program, as specified in the PARM parameter of the PGM command.

Shared files remain open if control is transferred to a program from a high-level language program or from the CL program that opened the files.

Restrictions: (1) This command is valid only within CL programs. (2) The user must have use authority or one of the data authorities for the program to which control is being transferred.

Required Parameters

PGM

Specifies the qualified name of the program that receives control from the program transferring control.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of the program that receives control.

Optional Parameters

PARM

Specifies the names of one or more CL variables that are passed to the program that is receiving control. The variables passed can only be parameters that were passed to the program currently transferring control. Up to 40 variables can be specified. Specify a CL variable name for each of the values being passed by the program transferring control. The parameter values are received in the receiving program in the order in which they were specified on the TFRCTL command that calls the program.

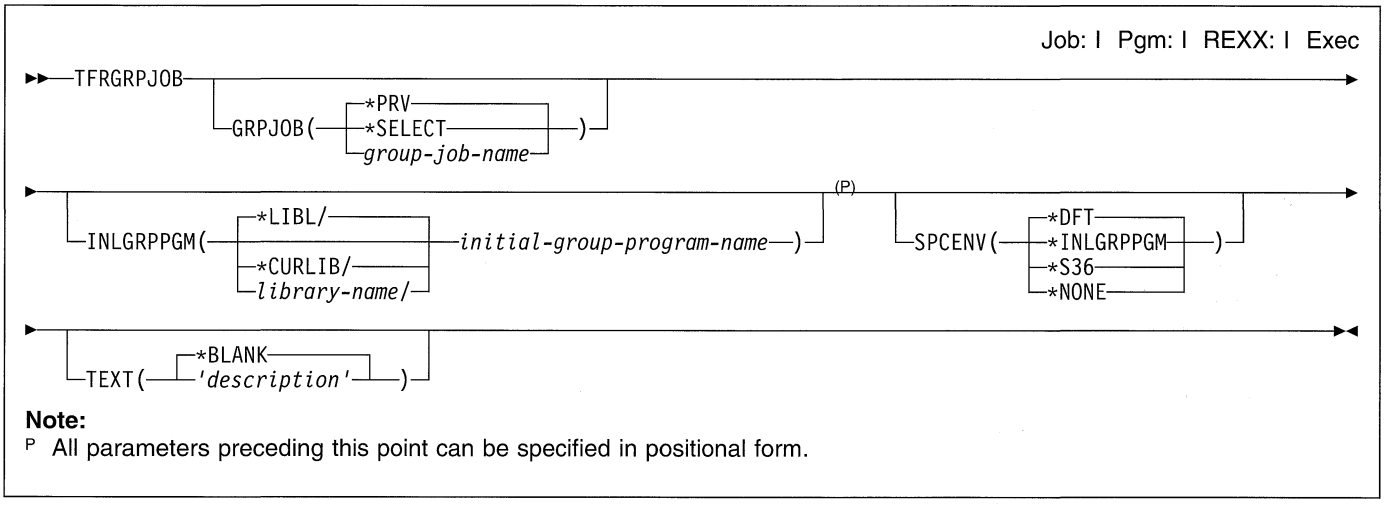
The value *N cannot be specified as a value being passed because a null value cannot be passed to another program.

Example

```
TFRCTL  PROGA  &PARM1
```

This command transfers control to the program PROGA and passes the parameter &PARM1 to it. The parameter &PARM1 must previously have been passed to the program issuing this command.

TFRGRPJOB (Transfer to Group Job) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Transfer to Group Job (TFRGRPJOB) command suspends the job that issued the TFRGRPJOB command and the group job specified by the GRPJOB parameter is resumed (if it already exists) or is created (if it does not exist). In both cases, control is transferred to the job specified by the GRPJOB parameter.

The job issuing the TFRGRPJOB command remains suspended until control is passed back to it and the job is resumed. A suspended job cannot do any processing until it is resumed. Suspended group jobs can regain control if another group job issues the TFRGRPJOB command to transfer to the suspended job, or if the active job in the group ends (and the suspended job was specified as the job to gain control).

If the work station message queue is in break or notify mode in the job that issued the TFRGRPJOB command, the message queue is set to the same mode in the group job that is transferred to. For example, if group job A has the work station message queue in break mode and transfers to group job B, the queue is in break mode in group job B. If group job B changes the queue to notify mode and transfers back to group job A, the queue is in notify mode in group job A.

If a user message queue is associated with the group (Change Group Attributes (CHGGRPA) command), it is handled like the work station message queue when one group job transfers to another group job.

Note: No message is sent for normal completion of the TFRGRPJOB command. Instead, the user may obtain a control code describing the circumstances under which the job regained control (along with the group job name and the job number of the previously active job) by using the RTVGRPA command. For more information about these control codes, see the Retrieve Group Attributes (RTVGRPA) command.

Optional Parameters

GRPJOB

Specifies the name of the group job to which control is transferred.

***PRV:** Control is transferred to the previously active job in the group. If the previously active job no longer exists, then the most recently active job in the group is resumed. This special value is valid only if there is another group job in the group.

***SELECT:** The group job selection display is shown. The user can choose which group job to transfer to or create a new group job and transfer to it.

group-job-name: Specify the group job name of the job to which control is transferred.

When a group job name is specified and that group job already exists, the group job is resumed. Whenever an already existing group job is resumed, the INLGRPPGM, SPCENV, and TEXT parameters are ignored. However, when a group job name is specified and that group job does not already exist, the INLGRPPGM parameter is required and must specify an existing program. When this is the case, a group job is started and control is passed to the new group job.

INLGRPPGM

Specifies the qualified name of the job's first group program. The INLGRPPGM parameter is valid only when a group job is created. If the group job being transferred to already exists, this parameter is ignored.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

initial-group-program-name: Specify the name of the first group program.

SPCENV

Specifies the environment in which the group job starts. This parameter is valid only when this command creates a group job. If control is transferring to an existing group job, this parameter is ignored.

***DFT:** The new group job starts in the environment specified in the user profile (the environment in which this command is running). The group job starts in the System/36 environment if one of the following is true:

- The System/36 environment is active in the job in which this command is running.
- The user profile specifies that the user runs in the System/36 environment, and the first program called in the group job is QCMD.

***INLGRPPGM:** The new group job starts in the environment determined by the first program called in the group job. If the first group program is QCMD, the special environment value in the user profile is used to determine the environment.

***S36:** The new group job starts in the System/36 environment.

***NONE:** The new group job starts in the AS/400 environment.

TEXT

Specifies text that briefly describes the group job. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

This text appears on the Group Job Selection display.

Note: This parameter only has meaning when a group job is created. If the group job being transferred to already exists, this parameter is ignored.

***BLANK:** Text is not specified.

'description': Specify no more than 50 characters of text, enclosed in apostrophes.

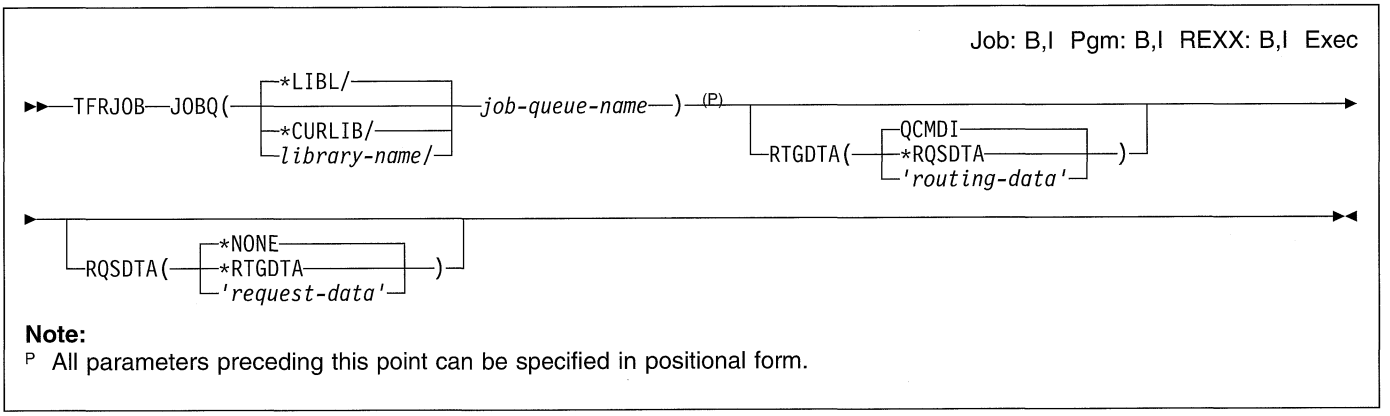
Example

```
TFRGRPJOB  GRPJOB(GROUPJ1)  INLGRPPGM(QGPL/PROGRAM1)
  SPCENV(*S36)
```

This command suspends running of the current job. If group job GROUPJ1 already exists, it is resumed at the point where it was suspended (the next high-level language command following the TFRGRPJOB request).

If group job GROUPJ1 does not exist, group job GROUPJ1 is created and runs the program QGPL/PROGRAM1 in the System/36 environment.

TFRJOB (Transfer Job) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Transfer Job (TFRJOB) command transfers a job to the specified job queue that is run in the subsystem in which that queue is active. The job that is transferred is the one in which this TFRJOB command is issued. The specified job queue is normally in a different subsystem than the one in which the job is currently. If the job being transferred is an interactive job, it is given the highest priority on the job queue. New routing data and request data can be specified for the job when it is transferred. If objects that were allocated to the previous routing step or files that were open in the previous routing step are needed in the new routing step, they must be allocated or opened again.

Note: Running this command in a batch job causes loss of spooled inline files because they cannot be accessed in the new routing step.

If the target subsystem is ended (by running the End Subsystem (ENDSBS) command, the End System (ENDSYS) command, or the Power Down System (PWRDWNSYS) command) while an interactive transferring job is on a job queue, the job is canceled as part of subsystem ending.

Because a Power Down System (PWRDWNSYS) command inhibits new jobs or routing steps from being started by any subsystem, a batch job transferred to a job queue (by the TFRJOB command) is not completed before the system is powered down. The temporary objects associated with a transferring job (such as the library list, the QTEMP library, and all objects in it) are destroyed during the PWRDWNSYS command, so that during a re-initial program load (IPL), the system is unable to restore the job to its previous state. During re-IPL, the system removes the job from the job queue and produces its job log.

Restrictions: The user must have read and add authority for the job queue and for the subsystem to which the job queue is allocated. If the job being transferred is an interactive job, the following restrictions apply:

- The job queue on which the job is placed must be associated with an active subsystem.
- The work station associated with the job must have a corresponding work station entry in the subsystem description associated with the new subsystem.
- The work station associated with the job must not have another job associated with it that has been suspended by means of the Sys Req (system request) key. The suspended job must be canceled before the Transfer Job command can be run.
- The job must not be a group job.
- The job must not be a communications batch job (started as a result of a program start request), unless it meets one of the following criteria:
  - It was started from an APPC communications device
  - The session on the communications device has ended

Required Parameters

JOBQ

Specifies the qualified name of the job queue to which the job is transferred.

| The name of the queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-queue-name: Specify the name of the job queue to which the job is transferred.

Optional Parameters

RTGDTA

Specifies the routing data used with this job description to start jobs. The routing data is used to determine the routing entry (in the subsystem description) that identifies the program in which the job runs.

QCMDI: This routing data matches a routing entry in the IBM-supplied QINTER subsystem description, which starts a routing step processed by the IBM-supplied control language processor, QCMD, in the QSYS library.

***RQSDTA:** Up to 80 characters of the request data, specified in the RQSDTA parameter of this command, are also used as the routing data for the routing step.

'routing-data': Specify the character string that is used as the routing data for starting the routing step. Up of 80 characters can be entered, enclosed in apostrophes, if necessary.

RQSDTA

Specifies the request data that is added to the end of the job's message queue for use by the new routing step.

***NONE:** No request data is placed in the job's message queue.

***RTGDTA:** The routing data specified in the RTGDTA parameter is also placed at the end of the job's message queue.

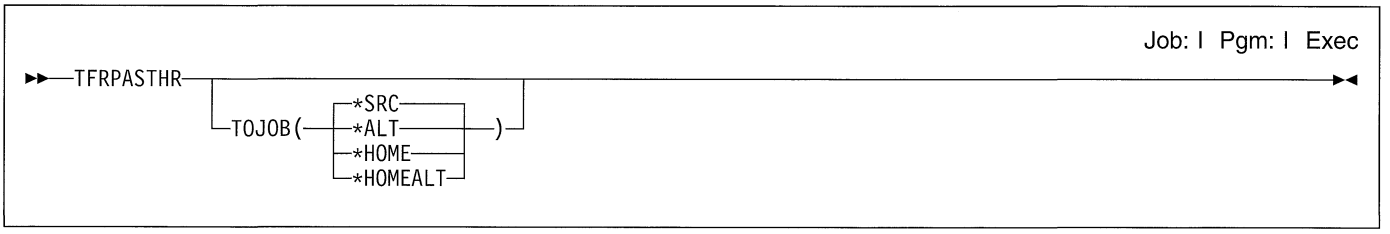
'request-data': Specify the character string that is placed at the end of the job's message queue for use by the new routing step or some subsequent routing step in the job. Up to 256 characters can be entered, enclosed in apostrophes, if necessary. When a CL command is entered, it must be enclosed in single apostrophes, and where apostrophes would normally be used *inside* the command, double apostrophes must be used instead.

Example

```
TFRJOB  JOBQ(QSYS/QCTL)  RTGDTA(APPLICS)
```

This command transfers the job in which the command is entered to the QSYS job queue in the QGPL library. The job is routed using the routing data APPLICS. If the job is an interactive job, the job queue must be allocated by an active subsystem.

TFRPASTHR (Transfer Pass-Through) Command



Purpose

The Transfer Pass-Through (TFRPASTHR) command is used to transfer from a pass-through system to a source system. The TFRPASTHR command performs the same function as a System Request (SYS REQ) option 10, 11, 13, or 14, and is valid only on a target pass-through system.

Optional Parameters

TOJOB

Specifies the program that is given control when the user is transferred to the home system or the previous system.

***SRC:** The job on the current system is suspended, and control is transferred back to the program specified on the SRQ10PGM parameter of the Start Pass-Through (STRPASTHR) command on the previous system. When the specified program ends, the target system gets control.

***ALT:** The job on the target system is suspended, and control is transferred back to the alternate job on the previous system. When control is transferred, the Transfer Job (TFRJOB) command can be used to transfer from the alternate job to the original job, giving control to the target system. Otherwise, when the alternate job ends, the target system gets control.

***HOME:** The job on the target system is suspended, and control is transferred back to the program specified on the SRQ10PGM parameter of the Start Pass-Through (STRPASTHR) command on the home system. When the specified program ends, the target system gets control.

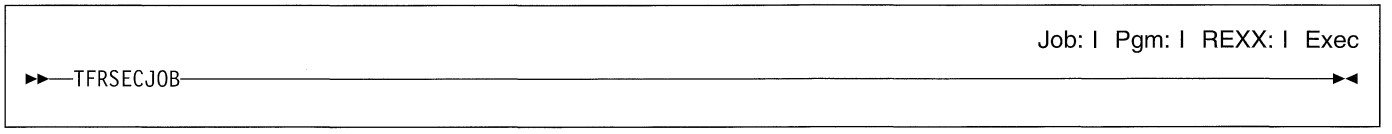
***HOMEALT:** The job on the target system is suspended, and control is transferred back to the alternate job on the home system. When control is transferred, the Transfer Job (TFRJOB) command can be used to transfer from the alternate job to the original job, giving control to the target system. Otherwise, when the alternate job ends, the target system gets control.

Example

```
TFRPASTHR  TOJOB(*HOME)
```

This command transfers control back to the source job on the home system.

TFRSECJOB (Transfer Secondary Job) Command



Purpose

The Transfer Secondary Job (TFRSECJOB) command creates a secondary interactive job at your work station, then transfers control between the primary and secondary jobs. The first time you issue this command, you receive the signon prompt for the secondary job. Once you sign on, a secondary job is created, allowing you to receive the basic working display of the new job (such as the command entry display). Your primary job remains suspended as long as you remain in your secondary job. The next time you issue the TFRSECJOB command, your current job is suspended, and you return to the first job at the point at which you left it.

When you sign off either job, you are automatically returned to the remaining job.

Note: This command performs the same function as option 1 on the System Request Menu.

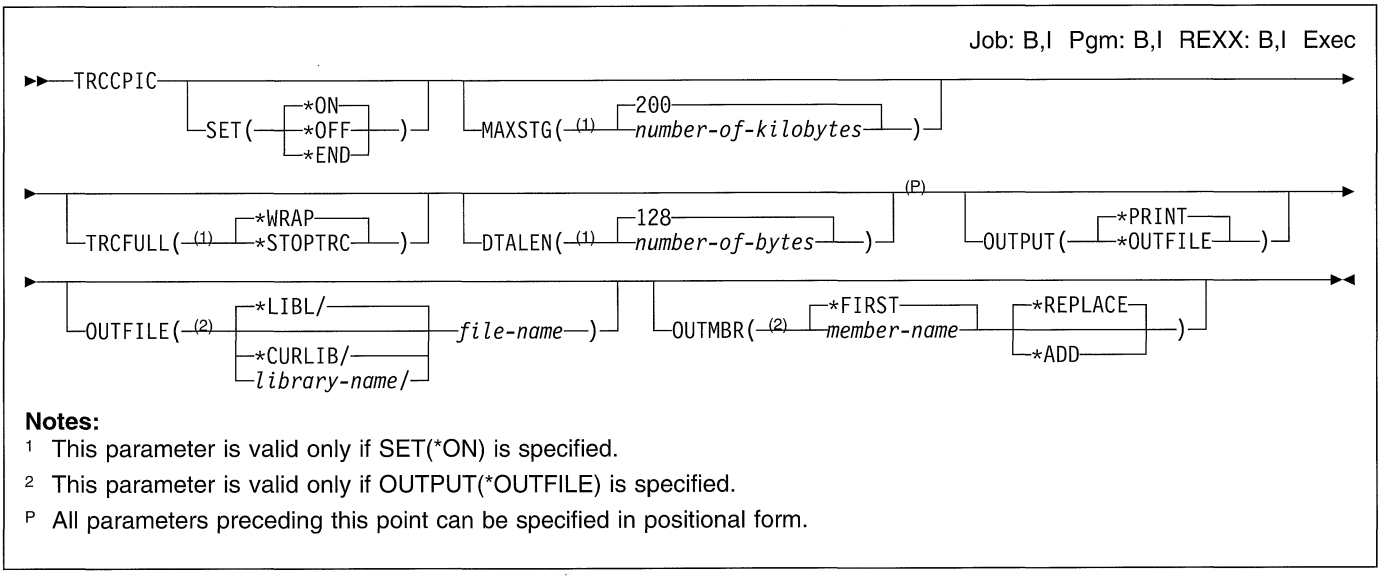
There are no parameters for this command.

Example

```
TFRSECJOB
```

The job that is currently running is suspended, and the SIGNON prompt is displayed. If a secondary job already exists, that job resumes running (the SIGNON prompt is not displayed).

TRCCPIC (Trace CPI Communications) Command



Notes:
1 This parameter is valid only if SET(*ON) is specified.
2 This parameter is valid only if OUTPUT(*OUTFILE) is specified.
P All parameters preceding this point can be specified in positional form.

Purpose

The Trace Common Programming Interface Communications (TRCCPIC) command controls tracing of all CPI Communications that occur in the job in which the command is entered. The command sets a trace on or off, and traces (1) CPI Communications calls issued by a program and (2) data that is sent and received.

As trace records are collected, they are stored in an internal trace storage area. When the trace is ended, the trace records can be directed to a spooled output file or a database physical file.

If the Start Service Job (STRSRVJOB) command is entered before the TRCCPIC command, the job that is traced is the one specified on the STRSRVJOB command. The trace output from the serviced job is returned to the servicing job after the trace is set off or after the serviced job has ended.

Restrictions: (1) The record format of the database output file must match the record format of the IBM-supplied output file, QACM0TRC. (2) The user must have specific authority from the security officer to use this command.

Optional Parameters

SET

Specifies whether a CPI Communications trace is started or ended.

***ON:** The trace is started. If the trace storage area becomes full, the action specified on the TRCFULL parameter is taken.

***OFF:** The trace is ended. No other trace information is recorded, and the current information is written to the spooled output file or a database file.

***END:** The trace ends. No other trace information is recorded and all current trace information is deleted. No output is generated.

MAXSTG

Specifies the maximum amount of storage (in kilobytes) used for the created trace records.

200: Up to 200KB of storage is used for trace records.

number-of-kilobytes: Specify the number of kilobytes of storage to use for trace records. Valid values range from 1 through 16000.

TRCFULL

Specifies the action taken when the maximum storage specified is full.

***WRAP:** When the trace storage area is full, new trace information is written over the old information, starting at the beginning of the storage area.

***STOPTRC:** When the trace storage area is full, no new trace information is saved.

DTALEN

Specifies the maximum length (in bytes) of user data that can be saved for each trace entry in the storage area. If the value specified is greater than the length of data received or sent across the communications line, only the actual data is traced. If the value specified is less than the data length received or sent, only the data length specified on this parameter is traced.

128: The maximum length of user data saved is 128 bytes.

number-of-bytes: Specify the maximum length of user data saved. Valid values range from 0 through 4096.

OUTPUT

Specifies whether the output from the command is printed with the job's spooled output or is directed to a database file. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

OUTFILE

Specifies the qualified name of the physical file to which the trace output is directed. If the file already exists, the system uses it. If the file does not exist, the system creates it. If the file is created, the text is "Output file for TRCCPIC."

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the physical file to which the trace output is directed.

OUTMBR

Specifies the name of the member in the physical file that receives the trace output. If the file is created by the system, a member is created with the name specified on this parameter. If the file exists but the member does not, a member with the specified name is created.

Element 1: Member to Receive Output

***FIRST:** The first member of the file specified on the OUTFILE parameter receives the trace output. If the file is created and *FIRST is specified, the name of the created member is the same as that of the file.

member-name: Specify the name of the member in the file that receives the trace output.

Element 2: Operation to Perform on Member

***REPLACE:** The new data replaces the existing data.

***ADD:** The new data is added to the file member at the end of the data already in the member.

Examples

Example 1: Starting Trace Operation

```
TRCCPIC  MAXSTG(350)  DTALEN(256)
```

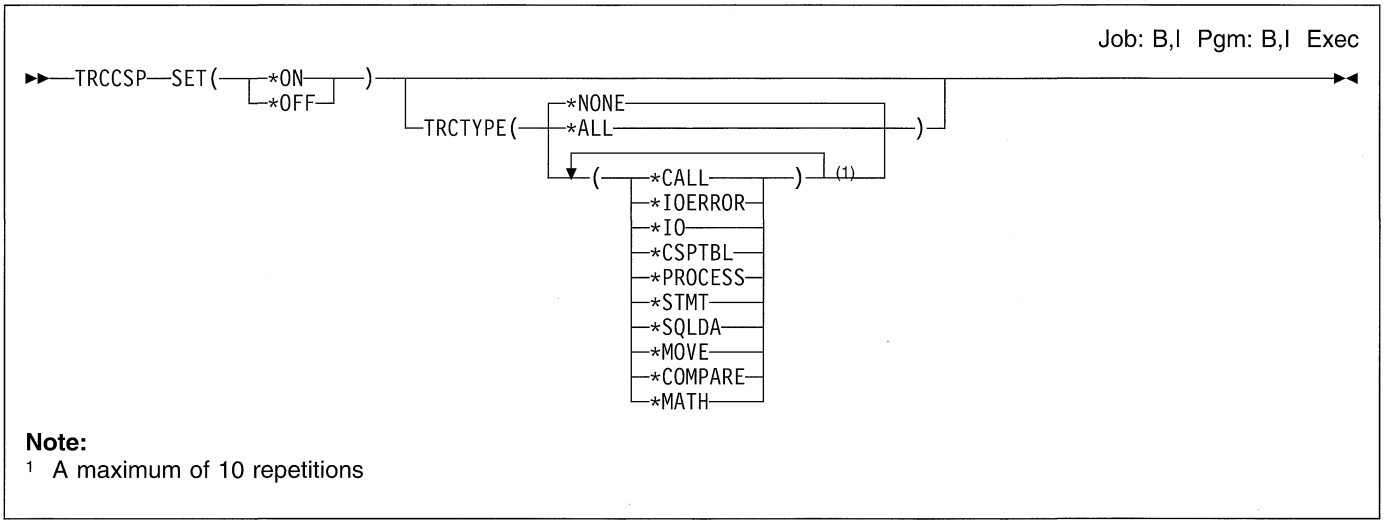
This command traces the CPI Communications calls of the current job. The trace file contains 350KB of storage and wraps to the beginning if that amount of storage is filled with trace records. In addition, this command traces up to 256 bytes of user data on each input/output operation.

Example 2: Stopping Trace Operation

```
TRCCPIC  SET(*OFF)  OUTPUT(*OUTFILE)
  OUTFILE(TRACELIB/CPICTRACE)
  OUTMBR(TRACEMBR)
```

This command stops the trace and directs the output to the database file CPICTRACE in library TRACELIB. The output is directed to the member TRACEMBR.

TRCCSP (Trace CSP/AE Application) Command



Note:
1 A maximum of 10 repetitions

Purpose

The Trace CSP/AE Application (TRCCSP) command starts and ends tracing of a Cross System Product/Application Execution (CSP/AE) application.

The spooled file QPAETRC, which is placed in the current job's output queue, will contain the trace output.

Required Parameters

SET

Specifies whether trace for CSP/AE applications is started or stopped.

***ON:** The trace for CSP/AE applications is started.

***OFF:** The trace for CSP/AE applications is stopped.

Optional Parameters

TRCTYPE

Specifies the valid trace options.

***NONE:** Tracing is prevented until the Attention key is pressed, after which the trace selections may be altered.

***ALL:** All of the trace selection flags (*CALL, *IOERROR, *IO, *CSPTBL, *PROCESS, *STMT, *SQLDA, *MOVE, *COMPARE, and *MATH) are used for tracing.

Up to 10 of the following trace options can be specified.

***CALL:** CALL and TRANSFER trace information is traced.

***IOERROR:** File I/O error information is traced.

***IO:** File I/O completion code information, including file I/O errors, is traced.

***CSPTBL:** Table object load information is traced.

***PROCESS:** Process names are traced.

***STMT:** Statement numbers are traced.

***SQLDA:** The SQLDA control block is traced.

***MOVE:** The contents of the source operand of CSP/AD move statements are traced.

***COMPARE:** The contents of the operands of CSP/AD comparison statements are traced.

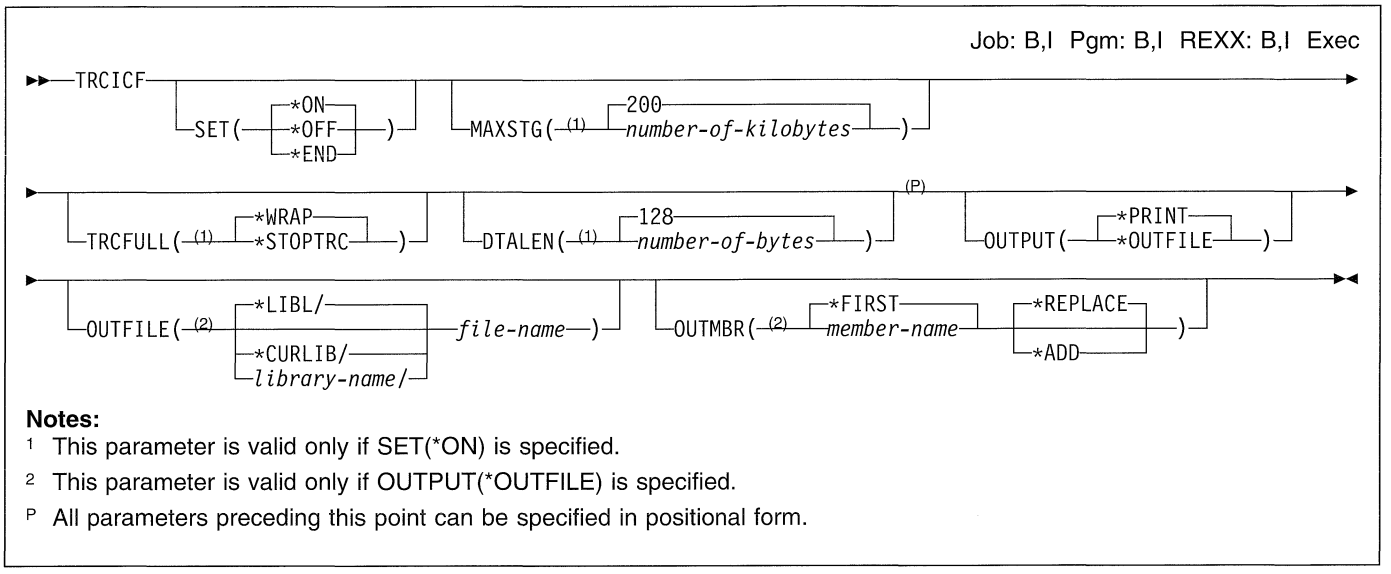
***MATH:** The contents of the operands of CSP/AD arithmetic statements are traced.

Example

```
TRCCSP  SET(*ON)  TRCTYPE(*ALL)
```

This command starts a trace environment for CSP/AE applications. All possible trace information is gathered for any CSP/AE application that runs after this command.

TRCICF (Trace ICF) Command



Notes:
1 This parameter is valid only if SET(*ON) is specified.
2 This parameter is valid only if OUTPUT(*OUTFILE) is specified.
P All parameters preceding this point can be specified in positional form.

Purpose

The Trace Intersystems Communications Functions (TRCICF) command controls tracing of all ICF I/O functions that occur in the job in which the command is entered. The command sets a trace on or off, and traces (1) ICF operations and functions issued by a program and (2) data that is sent and received.

As trace records are collected, they are stored in an internal trace storage area. When the trace is ended, the trace records can be directed to a spooled output file or a database physical file.

If the Start Service Job (STRSRVJOB) command is entered before the TRCICF command, the job that is traced is the one specified on the STRSRVJOB command. The trace output from the serviced job is returned to the servicing job after the trace is set off or after the serviced job has ended.

Restrictions:

1. The record format of the database output file must match the record format of the IBM-supplied output file, QAIFTRCF.
2. The user must have specific authority from the security officer to use this command.

Optional Parameters

SET
Specifies whether an ICF trace is started or ended.

***ON:** The trace is started. If the trace storage area becomes full, the action specified on the TRCFULL parameter is taken.

***OFF:** The trace is ended. No other trace information is recorded, and the current information is written to the spooled output file or a database file.

***END:** The trace ends. All trace information is deleted. No output is generated.

MAXSTG
Specifies the maximum amount of storage (in kilobytes) used for the created trace records.

200: Up to 200KB of storage is used for trace records.

number-of-kilobytes: Specify the number of kilobytes of storage to use for trace records. Valid values range from 1 through 16000.

TRCFULL
Specifies the action taken when the maximum storage specified is full.

***WRAP:** When the trace storage area is full, new trace information is written over old information, starting at the beginning of the storage area.

***STOPTRC:** When the trace storage area is full, no new trace information is saved.

DTALEN
Specifies the maximum length (in bytes) of user data that can be saved for each trace entry in the storage area. If the value specified is greater than the length of data received or sent across the communications line, only the actual data is traced. If the value specified is less than the data length received or sent, only the data length specified on this parameter is traced.

128: The maximum length of user data saved is 128 bytes.

number-of-bytes: Specify the maximum length of user data saved. Valid values range from 0 through 4096.

OUTPUT
Specifies whether the output from the command is printed with the job's spooled output or is directed to a

database file. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

OUTFILE

Specifies the qualified name of the physical file to which the trace output is directed. If the file already exists, the system uses it. If the file does not exist, the system creates it. If the file is created, the text is "OUTFILE created by TRCICF command." *EXCLUDE authority is assigned to users with no specific authority.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the physical file to which the trace output is directed.

OUTMBR

Specifies the name of the database file member to which the output is directed. If a member already exists, the system uses the second element of this parameter to determine whether the member is cleared before the new records are added. If the member does not exist and a member name is not specified, the system creates a member with the name of the output file specified on the OUTFILE parameter. If an output file member name is specified, but the member does not exist, the system creates it.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

Examples

Example 1: Starting Trace Operation

```
TRCICF  MAXSTG(350)  DTALEN(256)
```

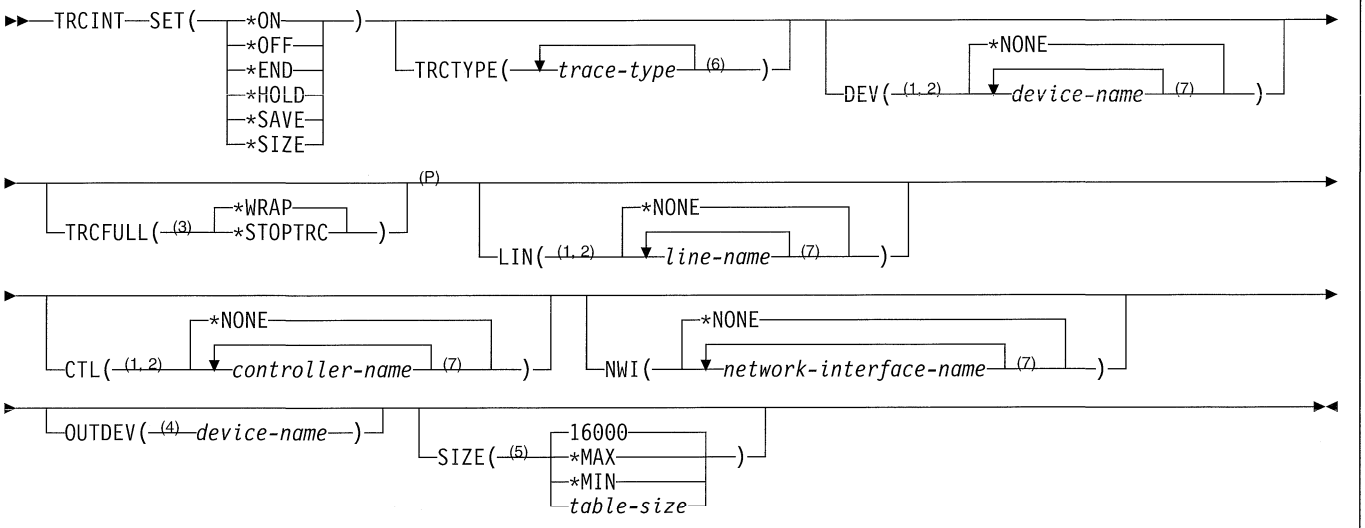
This command traces the ICF input/output operations of the current job. The trace file contains 350K of storage and wraps to the beginning if that amount of storage is filled with trace records. In addition, this command traces up to 256 bytes of user data on each input/output operation.

Example 2: Stopping Trace Operation

```
TRCICF  SET(*OFF)  OUTPUT(*OUTFILE)
  OUTFILE(TRACELIB/ICFTRACE)
  OUTMBR(TRACEMBR)
```

This command stops the trace and directs the output to the database file ICFTRACE in library TRACELIB. The output is directed to the member TRACEMBR.

TRCINT (Trace Internal) Command



Notes:

1 DEV, LIN, and CTL are valid only if TRCTYPE(010803) is specified.

2 A sum total of 16 devices (DEV), lines (LIN), and controllers (CTL) is allowed.

3 TRCFULL can be specified only if SET(*END) is specified.

4 OUTDEV is valid only if SET(*SAVE) is specified.

5 SIZE is valid only if SET(*SIZE) is specified.

6 A maximum of 50 repetitions

7 A maximum of 16 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Trace Internal (TRCINT) command is used primarily for problem analysis. It controls traces of internal events associated with the current job that occur at a level below the machine interface. Specific types of traces are started and stopped by using this command. Those that apply to devices can be limited to a particular device.

While previously started internal traces are being performed, additional internal traces can be started through this command. The output created by the trace is placed in internal storage used by the internal trace command. The records from the internal storage can be written to a spooled printer file or to a tape or diskette.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. The following user profiles have private authorities to use the command:
   - QPGMR
   - QSRV

Required Parameters

SET

Specifies whether internal tracing is started, stopped, ended, held, or saved. Also, the user can specify whether the trace table size is changed.

***ON:** The collection of internal trace records is started for the trace types specified in the TRCTYPE parameter. If the trace file already contains trace records, the new trace records are added to the file. If the file is full, the action specified in the TRCFULL parameter is taken.

***OFF:** Collection of internal trace records requested through previous TRCINT commands stops, and the records are written to the spooled printer file QPCSMPRT.

***END:** Internal tracing ends and the internal trace records are destroyed.

***HOLD:** Internal traces are stopped, and the collected internal trace records are held in internal storage. Held records can be printed later if another TRCINT command is entered that specifies SET(*OFF), or they can be put on tape or diskette if SET(*SAVE) is specified.

***SAVE:** Internal traces are stopped, and the internal trace records are written to the tape or diskette device specified by the OUTDEV parameter. The diskettes on which the records are written must be initialized in the basic exchange format (*DATA or *DATA2).

***SIZE:** The size of the trace collection table is changed. The new size is specified in the SIZE parameter.

Optional Parameters

TRCTYPE

Specifies the types of traces to start. The three groups of trace types are:

1. Component data trace codes: Active procedures are traced within the system.
2. Call data trace codes: Calls and returns done within the component are traced.
3. General trace codes: The System/38 instruction supervisor, multiprogramming, or task switching functions are traced.

This parameter can be specified only if SET(*ON) is specified. If a value other than *ON is specified on the SET parameter, TRCTYPE is ignored. Each trace type is identified by a 6-digit code; all 6 digits must be specified. Specify up to 50 trace types from those listed in Table 77.

DEV

Specifies the names of the devices for which the associated internal events are traced. This parameter can be specified only if TRCTYPE(010803) is specified.

***NONE:** No devices are traced by this command.

device-name: Specify the names of up to 16 devices for which the internal trace is started. The device names must be the same as the names specified in the associated device descriptions.

TRCFULL

Specifies whether the trace records wrap (replace the oldest records with new records) or stop tracing when the storage is full. This parameter can be specified only when SET(*END) is specified.

***WRAP:** When the trace storage is full, the trace wraps to the beginning. The oldest trace records are written over by new ones as they are collected.

***STOPTRC:** Tracing is stopped when the trace storage is full of trace records.

CTL

Specifies the names of the controllers for which the associated internal events are traced. This parameter is valid only if TRCTYPE(010803) is specified.

***NONE:** No controllers are traced by this command.

controller-name: Specify the names of up to 16 controllers for which the internal trace is started. The controller names must be the same as the names specified in the associated controller descriptions.

LIN

Specifies the names of the lines for which the associated internal events are traced. This parameter can be specified only if TRCTYPE(010803) is specified.

***NONE:** No lines are traced by this command.

line-name: Specify the names of up to 16 lines for which the internal trace is started. The line names must be the same as the names specified in the associated line descriptions.

NWI

Specifies the names of the network interfaces for which the associated internal events are traced. This parameter can be specified only if TRCTYPE(010803) is specified.

***NONE:** No network interfaces are traced by this command.

network-interface-name: Specify the names of up to 16 network interfaces for which the internal trace is started. The network interface names must be the same as those specified in the associated network interface descriptions.

OUTDEV

Specifies the device description name of the tape or diskette device on which the held trace records are written. This parameter must be specified if SET(*SAVE) is specified; otherwise it is ignored.

SIZE

Specifies the size of the data collection table. This parameter can be specified only when SET(*SIZE) is specified.

***16000:** The trace recording table is set to 16000 kilobytes.

***MAX:** The trace recording table is set to the maximum size of 998000 kilobytes.

***MIN:** The trace recording table is set to the minimum size of 128 kilobytes.

table-size: Specify the size of the trace recording table. Valid values range from 128 through 998000 kilobytes.

Table 77 (Page 1 of 2). Trace Code Table

Type of Trace	Component Trace Code	Call Data Trace Code	General Trace Code
System/38 instruction supervisor linkage (SVL) trace	--	--	030000
Multiprogramming level (MPL) trace	--	--	040000
Task switching trace	--	--	060000
Transaction trace	--	--	070000
APPN traces–(all)	012506	--	--
APPN control point manager	012501	052700	--
APPN control point presentation services	012504	053000	--
APPN directory services	012502	052800	--
APPN location manager	012505	053100	--
APPN management services transport	012507	--	--
APPN topology and routing services	012503	052900	--
Authority management	010900	050200	--
Auxiliary storage management	011101	--	--
Auxiliary storage management–detailed	011104	--	--
Commit management	011700	050300	--
Common class input/output manager (CCIOM)	011900	052100	--
Common functions	011200	050100	--
Communications answer manager (ISDN)	012800	--	--
Communications trace service function	012300	052500	--
Context management	011000	050400	--
Database management (events for all database files are traced)	010400	050500	--
Display station pass-through	010804	--	--
Environmental recording, editing and printing (EREP)	012200	052400	--
Error log	012100	052300	--
Event management	010600	050700	--
Exception management	010200	050600	--
Independent index management	011400	050800	--
Inter-process communications facility (IPCF)	012000	052200	--
Journal management	011600	050900	--
Link test service function	012400	052600	--
Load/dump (save/restore)	010801	052000	--
Machine observation	011300	051600	--
Machine services control point	010802	051500	--
Main storage management–details	011103	--	--
Main storage management–invocations	011102	--	--
Modula-2 runtime support	012700	--	--
Power management	012900	--	--
Process management	010500	051000	--
Program management	010300	051100	--
Queue management	010700	051800	--
Recovery management	--	051700	--
Resource management	010100	051200	--
Source/sink (device support) management	010803	051300	--
Source/sink IOM	--	051900	--
Space object management	011500	051400	--

Table 77 (Page 2 of 2). Trace Code Table

Type of Trace	Component Trace Code	Call Data Trace Code	General Trace Code
Storage management–(all)	011105	--	--
Virtual terminal manager	013000	--	--

Examples

Example 1: Starting Component Data Traces and Call Traces

```
TRCINT  SET(*ON)  TRCTYPE(010100 010400 050500 051200)
```

This command starts component data traces and call traces for resource management and database. Database operations associated with database files are used to collect component data trace records.

Example 2: Tracing Lines and Controllers

```
TRCINT  SET(*ON)  TRCTYPE(010803)  DEV(WS1 WS2 WS3)
LIN(L1 L2)  CTL(C1 C2)
```

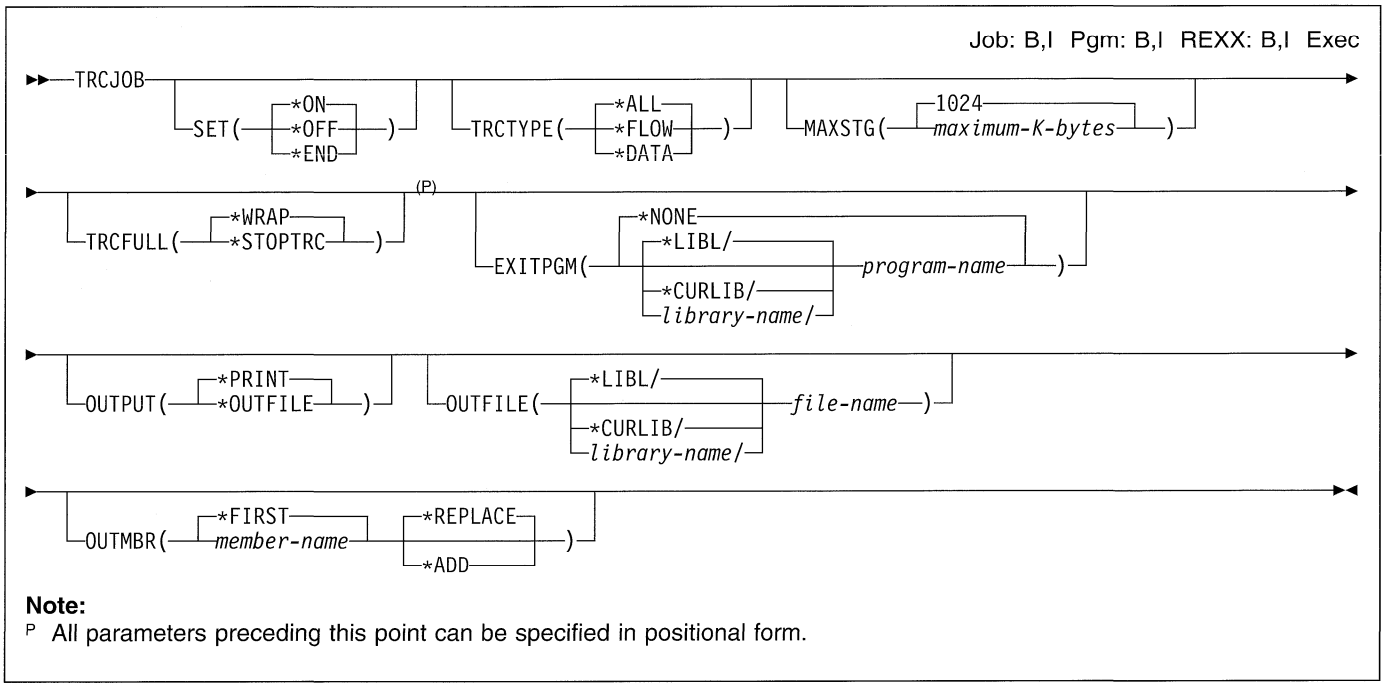
This command starts component data traces for source/sink management (device support) operations involving the devices WS1, WS2, and WS3, lines L1 and L2, and controllers C1 and C2.

Example 3: Stopping Traces and Clearing Trace Storage

```
TRCINT  SET(*END)  TRCFULL(*STOPTRC)
```

This command stops all traces and clears the trace storage. When TRCINT is used again, tracing stops when the trace storage becomes full.

TRCJOB (Trace Job) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Trace Job (TRCJOB) command controls traces of program calls and returns that occur in the current job or in the job being serviced as a result of the Start Service Job (STRSRVJOB) command directed to that job. This command, which sets a trace on or off, can trace module flow, OS/400 system data acquisition (including CL command traces), or both.

As the trace records are collected, they are stored in an internal trace storage area. When the trace is ended, the trace records can be written to a spooled printer file, QPSRVTRC. The trace records can also be directed to a database output file.

If the Start Service Job (STRSRVJOB) command is entered before the TRCJOB command, the job that is traced is the one identified by the STRSRVJOB command. The trace output from the serviced job is returned to the servicing job after the trace is set off or after the serviced job has ended.

Restrictions:

1. The record format of the database output file must match the record format of the IBM-supplied output file QATRCJOB.
2. The number of trace records processed between the start and end of the trace must not exceed one million.
3. This command is shipped with public *EXCLUDE authority.
4. The following user profiles have private authorities to use the command:
   - QPGMR
   - QSRV
   - QSRVBAS
   - QSYSOPR
   - QRJE

Optional Parameters

SET

Specifies whether the collection of trace records starts or stops.

***ON:** The collection of trace records is started.

***OFF:** The collection of trace records is stopped, and the trace records are written to the spooled printer file.

***END:** The collection of trace records is stopped, and all existing trace records are deleted. No spooled printer file is created.

TRCTYPE

Specifies the type of trace data to be stored in a trace file.

***ALL:** All the trace data collected is stored in trace records. This includes tracing the flow of control and the trace data itself.

***FLOW:** The flow of control is traced when programs are called and when they return control.

***DATA:** The data that is provided at predefined trace points within the OS/400 system is stored in trace records. This includes trace records for the CL commands that have run.

MAXSTG

Specifies the maximum amount of storage in kilobytes (K) used for collected trace records.

1024: Up to 1024 K-bytes of storage is used.

maximum-K-bytes: Specify the maximum amount of storage, in K-bytes, used to store trace records (one K equals 1024 bytes).

TRCFULL

Specifies whether the trace records wrap (replace oldest records with new records) or whether the trace stops when all of the storage specified by the MAXSTG parameter has been used.

***WRAP:** When the trace file is full, the trace wraps to the beginning. The oldest trace records are written over by new ones as they are collected.

***STOPTRC:** Tracing stops when the trace file is full of trace records.

EXITPGM

Specifies the qualified name of a user-written program that is given control just prior to the collection of each trace record.

Note: Items being traced can be missed when an exit program is used. Do not use an exit program if you do not want to risk losing a trace record.

***NONE:** No user-written program is called.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

program-name: Specify the name of the user-written program called before each trace record is collected. This program can examine the trace record passed to it as a parameter, and alter the first two characters. If it replaces the first two characters in the trace record with blanks or binary zeros, the entire trace record is suppressed.

There are four types of trace records: flow, data, suspend, and resume. The type is identified in the first byte of the record. The format of the flow record is:

Bytes	Description
1	Trace record type: X'00' = Flow
2	Flow type: X'01' = Call X'02' = Transfer control X'03' = Invoke an event handler X'04' = Invoke an exception handler X'05' = Invoke an internal or branch point exception handler X'06' = Return from an internal exception handler X'07' = Invocation exit X'08' = Return X'09' = Invocation terminated due to "resignal exception" X'0A' = Return from an external exception handler X'0B' = Terminate phase termination X'0C' = Terminate process due to unhandled exception X'0E' = Invocation terminated X'0F' = Cancel invocation
3-10	Timestamp
11-20	Name of program
21-30	Name of library the program is in
31-32	Invocation level (binary)
33-34	Number of successive records deleted by exit program (binary)
35-36	Entry Machine Interface instruction number (hex)
37-38	Exit Machine Interface instruction number (hex)
39-42	CPU time used
43-46	Number of data base pages read
47-50	Number of non-data base pages read
51-54	Number of pages written
55-56	Number of transfers to wait state
57-66	Name of module
67-76	Name of library the module is in
77-332	Name of procedure

There are four types of data records: data management, message handler, command analyzer, and generic. The data record type is identified in the second byte of the record. The format of the data management data record is:

Bytes	Description
1	Trace record type: X'01' = Data
2	Data source: X'01' = Open file X'02' = Close file X'0B' = Acquire device X'0C' = Release device
3-10	Timestamp
11-20	Internal file name
21-30	Actual file name (will be different than the internal file name if the file is overridden)
31-40	Library the actual file is in (*N is the file is an inline spool file)
41-50	For a device file, this is the device description name of the first device in the file For a data base file, this is the name of the member For an inline spooled file, this is *N
51-60	Program device name of the device in the device file

The format of the message handler data record is:

Bytes	Description
1	Trace record type: X'01' = Data
2	Data source: X'03' = SNDMSG X'04' = Reply to SNDMSG X'05' = Default exception handler X'06' = Send status message
3-10	Timestamp
11-11	Trace record type
12-18	Message id
19-19	Message type
20-21	Message severity (binary)
22-31	Name of message queue or program message queue
32-33	Receiving invocation number (binary)
34-289	Name of procedure
290-299	Name of module
300-309	Name of library
310-310	Message queue type

The format of the command analyzer data record is:

Bytes	Description
1	Trace record type: X'01' = Data
2	Data source: X'07' = Command Processor X'08' = Call program X'0A' = Control Language compiler Other values may appear, but are not assigned
3-10	Timestamp
11-20	Command name
21-30	Library the command is in
31-40	Command processing program
41-50	Library the command processing program is in

The format of the generic data record is:

Bytes	Description
1	Trace record type: X'01' = Data
2	Data source
3-10	Timestamp
11-310	Data

The format of the suspend record is:

Bytes	Description
1	Trace record type: X'02' = Suspend
2	Data source
3-10	Timestamp

The format of the resume record is:

Bytes	Description
1	Trace record type: X'03' = Resume
2	Data source
3-10	Timestamp

This program can also call any of the service dump commands and must return control when it completes its task. If the user-written program runs other OS/400 system commands, the output from those commands is associated with the job in which they are run. The dump and trace commands associate their output data either with the job that enters the dump and trace commands or with the job being serviced as the result of the STRSRVJOB command.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

OUTFILE

Specifies the physical database file to which the trace output is directed. If the output file already exists, the system tries to use it. Records replace existing data in the file member. If the output file does not exist, the system creates a database physical file (with the name specified in the OUTFILE parameter) in the specified library. A member is created for the file with the name specified in the OUTMBR parameter. If the file is created, the text is "OUTFILE created by TRCJOB command," and *EXCLUDE authority is given to users with no specific authority.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

file-name: Specify the name of the file to receive the trace output.

OUTMBR

Specifies the name of the database file member to which the output is directed. If a member already exists, the system uses the second element of this parameter to determine whether the member is cleared before the new records are added. If the member does not exist and a member name is not specified, the system creates a member with the name of the output file specified on the OUTFILE parameter. If an output file member name is specified, but the member does not exist, the system creates it.

Element 1: Member to Receive Output

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Element 2: Operation to Perform on Member

***REPLACE:** The system clears the existing member and adds the new records.

***ADD:** The system adds the new records to the end of the existing records.

Examples

Example 1: Tracing Flow of Control

```
TRCJOB  TRCTYPE(*FLOW)  MAXSTG(40)
```

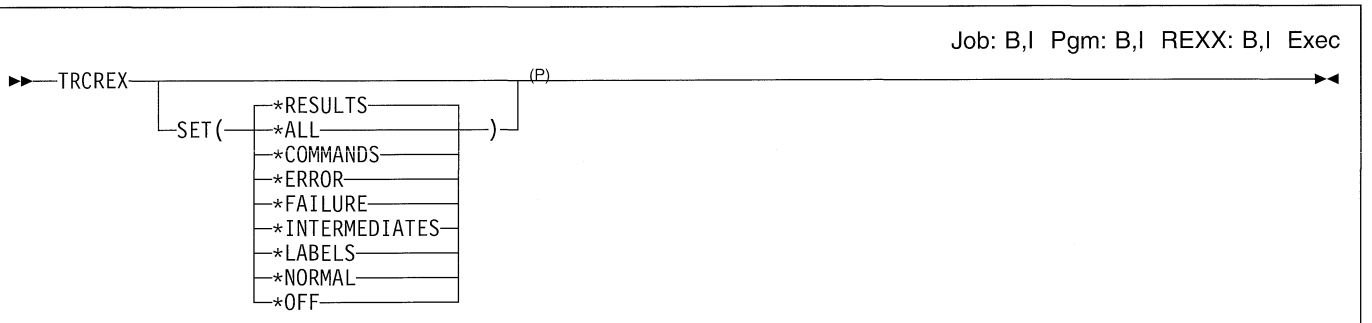
This command traces the module flow of the current job. Trace records are collected for each program call and return that occurs in the job. The trace file contains 40K of storage and wraps (oldest records are replaced by new records) if that amount of storage is filled with trace records.

Example 2: Stopping the Trace Operation

```
TRCJOB  SET(*OFF)  OUTPUT(*OUTFILE)  OUTFILE(QGPL/TRCJOB)
  OUTMBR(TRCDTA)
```

This command stops the trace and directs the output to the database file QGPL/TRCJOB. The output is directed to the member TRCDTA.

TRCREX (Trace REXX) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Trace REXX (TRCREX) command allows the user to turn the REXX interpreter tracing feature on or off from the command entry level or from the CL programming level.

Optional Parameters

SET

Specifies the initial trace setting for the next REXX procedure that is run. This setting remains in effect unless changed through the REXX Trace instruction.

***RESULTS:** All clauses are traced before processing. The final results of an expression evaluation are traced. The values that were assigned during the PULL, ARG, and PARSE instructions are also shown. Tracing operates as if the Trace instruction is issued from within the REXX procedure.

***ALL:** All clauses are traced before processing. Tracing operates as if the Trace instruction is issued from within the REXX procedure.

***COMMANDS:** All host commands are traced before processing and any error return code is shown. Tracing operates as if the Trace instruction is sent from within the REXX procedure.

***ERROR:** Host commands resulting in an error return code are traced after processing. Tracing operates as if the Trace instruction is sent from within the REXX procedure.

***FAILURE:** Host commands resulting in a failure are traced after processing with the return code from the command. Tracing operates as if the Trace instruction is sent from within the REXX procedure.

***INTERMEDIATES:** All clauses are traced before processing. Intermediate results during evaluation of expressions and substituted names are also traced. Tracing operates as if the Trace instruction is sent from within the REXX procedure.

***LABELS:** Labels passed during processing are traced. Tracing operates as if the Trace instruction is sent from within the REXX procedure.

***NORMAL:** Failing host commands are traced after processing. Tracing operates as if the Trace instruction is sent from within the REXX procedure.

***OFF:** Nothing is traced. Tracing operates as if the Trace instruction is sent from within the REXX procedure.

Examples

Example 1: Tracing Host Commands

```
TRCREX   SET(*COMMANDS)
```

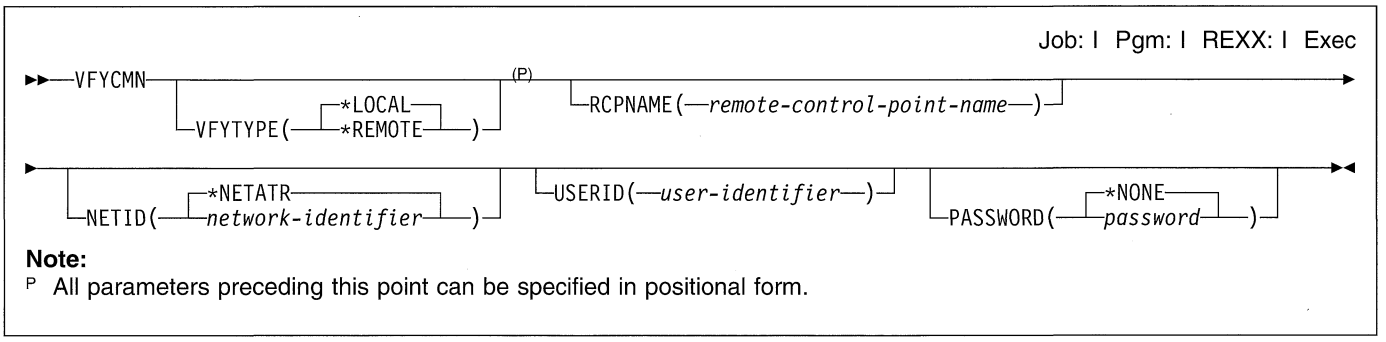
This command causes all commands in by the REXX procedure to be shown before they are to be run.

Example 2: Tracing Failing Host Commands

```
TRCREX   SET(*NORMAL)
```

This command causes all commands that result in a FAILURE condition to be shown. This command shows the normal setting for the REXX tracing operation.

VFYCMN (Verify Communications) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Verify Communications (VFYCMN) command shows the *Select a Line to Test* display allowing the user to make sure that communications equipment is operating properly.

Depending on the user's system configuration, the following tests can be run:

- Link
- Local modem
- Remote modem
- Cable
- Communications input/output adapter
- Link Problem Determination Aid-2 (LPDA-2)

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Optional Parameters

VFYTYPE
Specifies whether local or remote communications hardware is checked to verify that it is operating properly.

Note: The AS/400 Systems Management Utilities (5738SM1) licensed program must be installed to do remote analysis.

***LOCAL:** Communications hardware is checked to verify that it is operating properly on the local AS/400 system.

***REMOTE:** Communications hardware is checked to verify that it is operating properly on another AS/400 system that is enrolled as a service requester.

RCPNAME
Specifies the remote control point name for the service requester system where the remote verification is done.

Note: This parameter is valid only when VFYTYPE(*REMOTE) is specified.

NETID
Specifies the network identifier (ID) of the service requester system where the remote verification is done.

Note: This parameter is valid only when VFYTYPE(*REMOTE) is specified.

***NETATR:** The network ID of the service provider is used.

network-identifier: Specify the network ID of the service requester system where the remote verification is done.

USERID
Specifies the user identifier (ID) used to access the remote system.

Note: This parameter is valid only when VFYTYPE(*REMOTE) is specified.

PASSWORD
Specifies the password used to access the remote system.

Note: This parameter is valid only when VFYTYPE(*REMOTE) is specified.

***NONE:** No password is needed to access the remote system because the remote system has a security level of 10.

password: Specify the password needed to access the remote system.

Examples

Example 1: Showing Select a Line to Test Display

```
VFYCMN
```

This command shows the Select a Line to Test display.

Example 2: Checking a Remote System

```
VFYCMN  VFYTYPE(*REMOTE)
```

This command shows the display which prompts for the remaining values of the command. After you specify the appropriate values, remote analysis begins.

Example 3: Accessing a Remote System Using a Password

```
VFYCMN  VFYTYPE(*REMOTE)  RCPNAME(RCH38377)
  USERID(JON)  PASSWORD
```

This command shows the display which prompts for the remaining values of the command. After you specify the appropriate values beyond the ones specified on the command example, remote analysis begins.

Example 4: Accessing a Remote System Without a Password

```
VFYCMN  VFYTYPE(*REMOTE)  RCPNAME(RCH38377)
  USERID(JON)
```

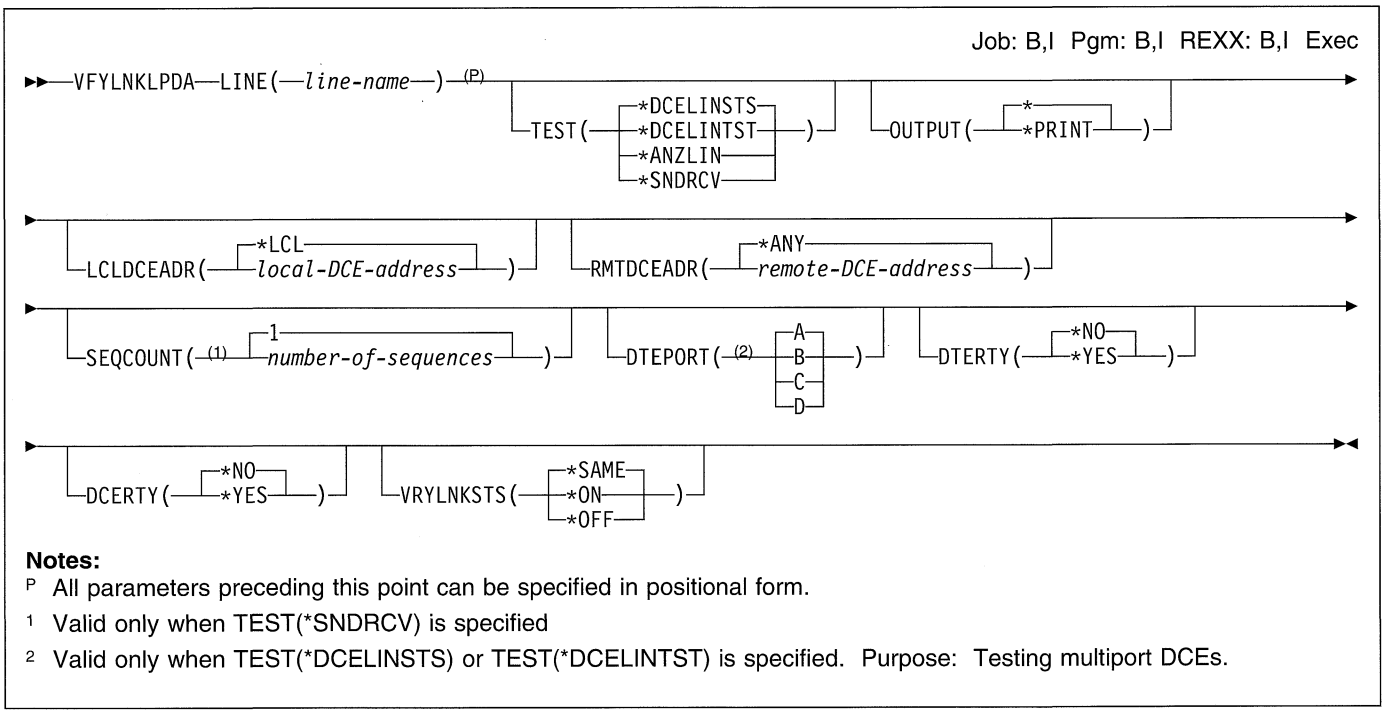
This command is similar to the preceding example except that the PASSWORD parameter is not specified. The same prompt display is shown, however, the system assumes that the remote system has a security level of 10, that is, it does not use passwords. Another prompt display appears after this command is specified. After the user specifies the appropriate values on this display, remote analysis begins.

Example 5: Checking a Local System

```
VFYCMN  VFYTYPE(*LOCAL)
```

This command begins analysis on the local device. The remaining parameters do not appear on the display.

VFYLNKLPDA (Verify Link Supporting LPDA-2) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 Valid only when TEST(*SNDRCV) is specified

2 Valid only when TEST(*DCELINSTS) or TEST(*DCELINTST) is specified. Purpose: Testing multiport DCEs.

Purpose

The Verify Link Supporting LPDA-2 (VFYLNKLPDA) command allows the user to run a Link Problem Determination Aid-2 (LPDA-2) test and then receive the results in a format specified by the user.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Required Parameters

LINE

Specifies the name of the line of the link to be tested.

Optional Parameters

TEST

Specifies which test to run.

***DCELINSTS:** Checks the status of the line and data circuit-terminating equipment (DCE).

***DCELINTST:** Tests the line and the data circuit-terminating equipment (DCE).

***ANZLIN:** Analyzes the line. This special value is valid only for analog lines.

***SNDRCV:** Tests sending and receiving capabilities.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

LCLDCEADR

Specifies the hexadecimal address of a local DCE. By convention, bits 4 through 7 of this byte indicate the Link Segment Level (LSL) of the local DCE and the remote DCE. Bits 0 through 3 are used to uniquely define a local DCE among several DCEs on the same LSL. The address is set on the local DCE during configuration and must follow this convention.

Note: '00'X is not a valid address for a local DCE.

***LCL:** '01'X, the address for the local DCE on LSL 1, is used.

local-DCE-address: Specify the address of the local DCE. Valid values range from X'01' through X'FB'.

RMTDCEADR

Specifies the hexadecimal address of a remote DCE.

This parameter must be specified if the user is testing a multipoint link.

***ANY:** 'FD'X, the global address of a remote DCE, is used. If a remote DCE is not idle, it responds to the 'FD'X address, regardless of its previously configured address.

Note: Multipoint tributary DCEs will not respond to an address of *ANY.

remote-DCE-address: Specify the address of the remote DCE. Valid values range from X'01' through X'FB'.

SEQCOUNT
Specifies the number of sequences to transmit during a send and receive test. A sequence is a group of 16 blocks. The block length is determined depending on how a DCE is configured.

1: One sequence is transmitted.

number-of-sequences: Specify the number of sequences to transmit. Valid values range from 1 through 3.

DTEPORT
Specifies the data terminal equipment (DTE) port on the remote DCE. The DTE status for this port is returned to the user. This parameter is valid only when working with line status and line testing of DCEs. In addition, this parameter is only useful for multiport DCEs.

A: The A-port is used.

B: The B-port is used.

C: The C-port is used.

D: The D-port is used.

DTERTY
Specifies whether the user is attempting to retry a test request from the system DTE to a local DCE due to a bad response or no response received from a local DCE.

***NO:** This is not a DTE retry.

***YES:** This is a DTE retry.

DCERTY
Specifies that the local DCE should retry a link operation to a remote DCE if a bad response or no response is received from the remote DCE.

***NO:** A DCE retry is not performed.

***YES:** A DCE retry is performed, if necessary.

VRYLNKSTS
Specifies whether the link is varied on or varied off following a completed test.

After running a test on a manually switched link, the link should be left varied on to allow further information to be received on the same connection. If a switched link is varied off, the connection may fail and no further analysis can be performed.

***SAME:** The value does not change.

***ON:** The link remains varied on after testing has completed.

***OFF:** The link is varied off after testing has completed.

Examples

Example 1: Checking Line Status

```
VFYLNKLPDA  LINE(LINE1)  DTEPORT(B)
```

This command retrieves the DCE line status from synchronous data link control (SDLC) line, LINE1, and displays the status. The remote DCEs DTE line connection status of port B is returned if the user is verifying a multiport DCE. An error message will be returned if the remote DTE has only a single port, for example, port A. The default VRYLNKSTS(*SAME) causes the line named LINE1 to return to the status prior to the test.

Example 2: Analyzing a Line

```
VFYLNKLPDA   LINE(LINE2)  TEST(*ANZLIN)
  OUTPUT(*PRINT)  LCLDCEADR(02)  VRYLNKSTS(*ON)
```

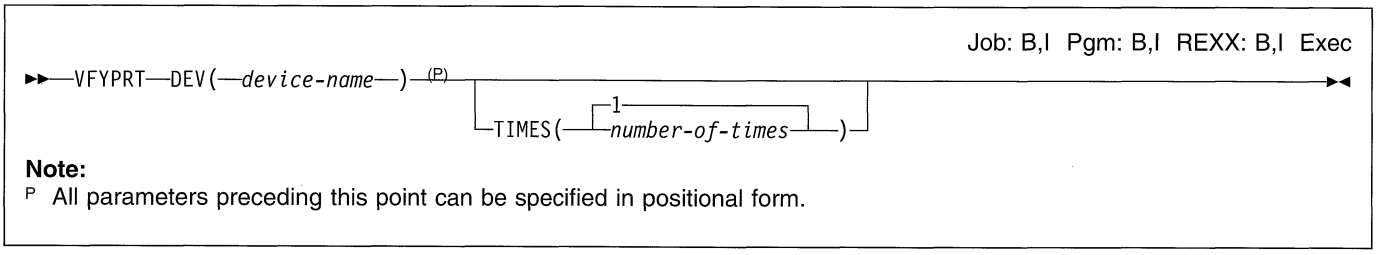
This command analyzes the SDLC line, LINE2. The second LSL is used; the lower four bits of the local DCE address (LCLDCEADR) are 2. The results are sent to a spooled file. After the test, LINE2 remains varied on to allow for more testing.

Example 3: Testing Sending and Receiving Capabilities

```
VFYLNKLPDA   LINE(LINE3)  TEST(*SNDRCV)
  SEQCOUNT(3)  RMTDCEADR(21)  DCERTY(*YES)
```

This command tests the sending and receiving capabilities on the multipoint line, LINE3. Three sequences of 16 blocks are sent between the local (control) DCE and the remote (tributary) DCE with the address of X'21'. If the local DCE fails to receive a response on the first attempt, the local DCE will retry this command to the remote DCE.

VFYPRT (Verify Printer) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Verify Printer (VFYPRT) command runs the 3287, 3812 SCS, 3812 IPDS, 4210, 4214, 4224, 4234, 4245, 5219, 5256, 5262, 5524, 5525, 5553, and 5583 Printers by causing them to print a test pattern a specified number of times.

Restrictions:

1. This command does not support advanced printer function printers that are configured with AFP(*YES) specified in their printer device descriptions. (Note that some advanced function printers, such as the 3820, 3827, and 3855 Printers, can be configured only in this manner.) This means that this command *can* exercise both *nonadvanced* function printers and advanced function printers, such as the 3812 and 3816 Printers, that have AFP(*NO) specified in their device descriptions.
2. This command is shipped with public *EXCLUDE authority.
3. The following user profiles have private authorities to use the command:
   - QPGMR
   - QSRV
   - QSRVBAS

Required Parameters

DEV

Specifies the name of the printer on which to run the test pattern. The device name must be the same as that specified in the device description for the printer.

Optional Parameters

TIMES

Specifies the number of times that the specified printer prints the test pattern.

1: The test pattern is printed once.

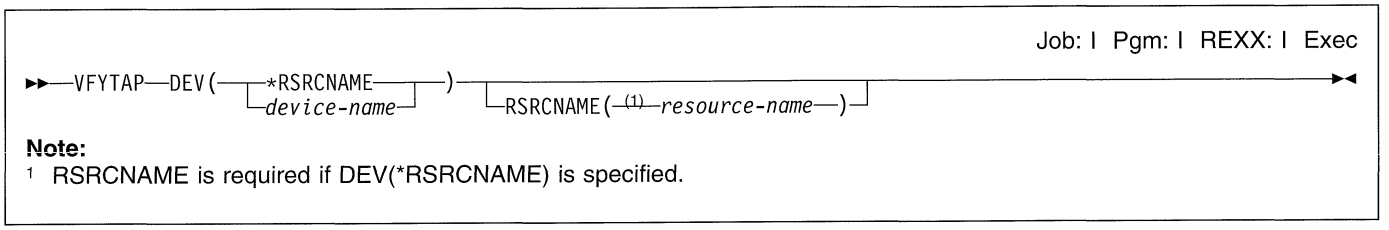
number-of-times: Specify a value for the number of times to print the test pattern.

Example

```
VFYPRT   DEV(PRTR3)  TIMES(15)
```

This command causes printer PRTR3 to print a test pattern 15 times.

VFYTAP (Verify Tape) Command



Purpose

The Verify Tape (VFYTAP) command verifies whether a specified tape unit is operating.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Required Parameters

DEV
Specifies the name of the tape unit whose operation is being verified.

***RSRCNAME:** The resource name of the tape unit whose operation is being verified is used.

device-name: Specify the name of the tape unit whose operation is being verified.

Optional Parameters

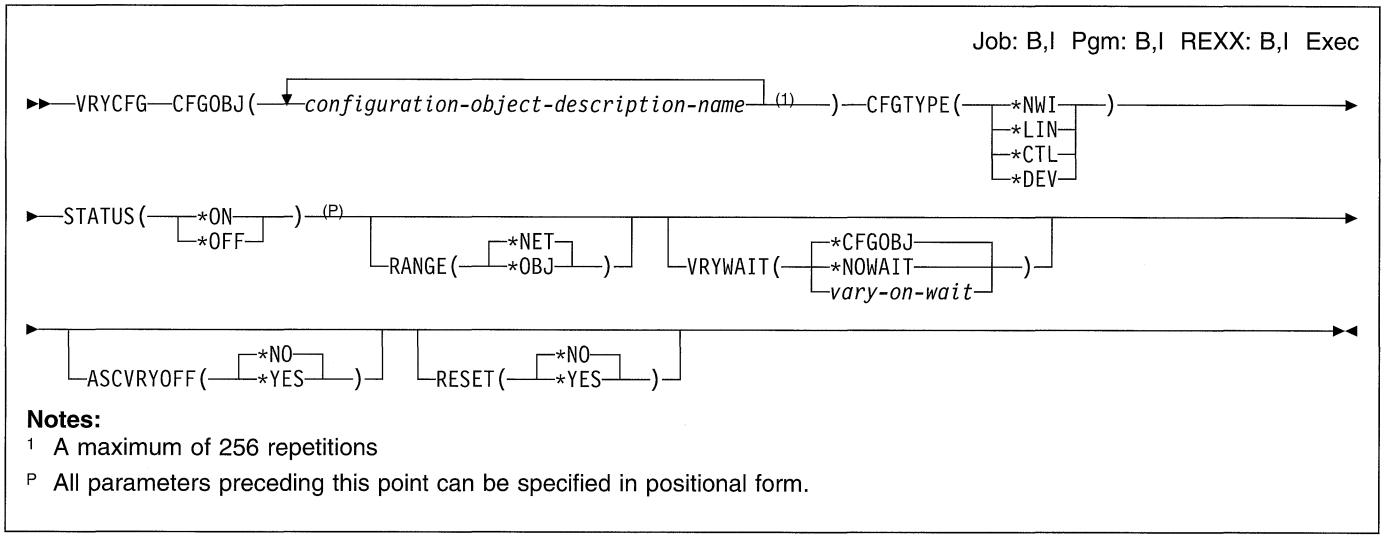
RSRCNAME
Specifies the resource name of the tape unit whose operation is being verified.

Example

```
VFYTAP  DEV(TAP3)
```

This command verifies whether the tape unit named TAP3 is working.

VRYCFG (Vary Configuration) Command



Notes:

1 A maximum of 256 repetitions

P All parameters preceding this point can be specified in positional form.

Purpose

The Vary Configuration (VRYCFG) command varies configuration objects on or off and optionally resets the input/output processor (IOP) associated with the specified objects. The VRYCFG command is used to vary on or off one or more configuration objects with the capability of also varying on the downline attached configuration objects. The configuration object types that can be varied on or off are network interface, line, controller, and device. This command applies to all lines, controllers, and devices on the system.

Downline attached objects can be varied on or off along with the specified object by specifying the value *NET on the range parameter (RANGE(*NET)). Downline attached objects of a network interface description are all the attached lines and all controllers attached to the lines and all the devices attached to the controllers. Downline attached objects of a line are all the attached controllers and all the devices attached to the controllers. Downline attached objects of a controller are all the attached devices. Devices do not have downline attachments. The RANGE parameter has no effect when varying devices.

Varying on lines synchronously or asynchronously can be controlled by the VRYWAIT parameter. This applies only to Ethernet, token-ring, X.25, or switched IDLC, SDLC, BSC, Async line descriptions, or network interface descriptions. The value specified for the VRYWAIT parameter determines how long the system waits until either the line or network interface is varied on before completing the VRYCFG command, or until the timer expires.

The time to vary on a line is the time it takes to put tasks in place to manage the line, the time to activate the communications input/output processor (IOP), including download of the IOP microcode, the time to establish communications with the data circuit-terminating equipment (DCE) and other unique protocol set up tasks.

Line vary on time does not include telephone dialing time, however; a powered off modem may prevent vary on completion and cause the wait time to expire. If the timer expires, an informational message is sent to the QSYSOPR message queue. This is followed by the vary completion message.

The VRYCFG command can also be used to reset the IOP. An IOP can be a communications controller, local work station, or magnetic media controller. An IOP reset is valid only when varying on network interface descriptions, lines (except twinaxial data link controller (TDLC) lines), local work station controllers, tapes, and diskettes.

A line cannot be varied on:

- For IDLC lines, until the network interface description is varied on
- For switched lines, until a dial connection has been completed

A controller cannot be varied on:

- For leased (nonswitched) lines, if the line to which it is attached is varied off
- For switched lines, until a dial connection has been completed

A device cannot be varied on:

- If the controller to which it is attached is varied off. This does not apply to tape and diskette devices because they are not attached to a controller.

A network interface description cannot be varied off:

- Until all attached lines, controllers, and devices are varied off

A line cannot be varied off:

- Until all the attached controllers and devices are varied off

A controller cannot be varied off:

- If it is being used or allocated for use
- Until all the attached devices are varied off

A device cannot be varied off:

- If it is being used or allocated for use

Required Parameters

CFGOBJ
Specifies the name of the configuration object to be varied on or off. The name can have up to 10 characters.

CFGTYPE
Specifies the type of object to be varied on or off.

***NWI:** The network interface is varied on or off.

***LIN:** The line is varied on or off.

***CTL:** The controller is varied on or off.

***DEV:** The device is varied on or off.

STATUS
Specifies whether the system varies the object on or off.

***ON:** The object is varied on.

***OFF:** The object is varied off.

Optional Parameters

RANGE
Specifies which configuration elements are varied on or off.

Note: For devices: Because devices do not have downline attached objects, value *NET is not valid.

For switched lines: When varying on elements, value *NET is not valid. When varying off elements and specifying value *NET, the line and its downline attached objects are varied off.

For network interface descriptions: Specifying value *NET varies on all nonswitched attachments and varies off all nonswitched attachments.

***NET:** All downline attached configuration elements are varied on or off.

***OBJ:** Only the specified objects are varied on or off.

VRYWAIT
Specifies whether the line is varied on asynchronously or synchronously. For synchronous vary on, specifies how long the system waits for the vary on to complete.

Note: If the VRYWAIT parameter is specified on the VRYCFG command for a line description that is not Ethernet, token-ring, X.25, or switched SDLC, BSC, or Async, the parameter is accepted but ignored.

***CFGOBJ:** The VRYWAIT parameter value specified in the line description or network interface description is used.

***NOWAIT:** The system does not wait for vary on completion. The line or network interface is varied on asynchronously.

vary-on-wait: Specify the time (in seconds) to wait. Valid values range from 15 through 180. The system waits until the line is varied on, or until the specified time passes, before completing the Vary Configuration (VRYCFG) command.

Notes:

1. Specifying a wait time in the line description affects system IPL time, if ONLINE(*YES) is used, by the amount of time it takes to synchronously vary on the line or reach the wait-time value.
2. The time required to vary on a line is the time it takes to put tasks in place to manage the line, to activate the communications I/O processor (IOP) (including downloading the IOP model-unique Licensed Internal Code), and to establish communications with the data circuit-terminating equipment (DCE). Normal vary-on time ranges from 5 through 45 seconds, but can be longer, depending on the system, line protocol, and other factors.

ASCVRYOFF
Specifies whether the vary off is asynchronous. This parameter is not allowed when STATUS(*ON) is specified.

***NO:** The vary off is synchronous.

***YES:** The vary off is asynchronous.

RESET
Specifies whether the IOP associated with the object is reset.

***NO:** The associated IOP is not reset.

***YES:** The associated IOP is reset.

Examples

Example 1: Varying On the Network Interface and Attached Downline Attachments

```
VRYCFG  CFGOBJ(NWI1)  OJBTYPE(*NWI)  STATUS(*ON)
```

This command varies on the network interface and all attached downline attachments.

Example 2: Varying Off the Line and Attached Downline Objects

```
VRYCFG  CFGOBJ(LINE1)  OJBTYPE(*LIN)  STATUS(*OFF)
```

This command varies off the line and all attached downline objects. The RANGE parameter took the default value of *NET.

Example 3: Varying on the Controller

```
VRYCFG  CFGOBJ(CONTROLLER1)  OBJTYPE(*CTL)  STATUS(*ON)
  RANGE(*OBJ)
```

This command varies on only the controller.

Example 4: Varying on the Device

```
VRYCFG  CFGOBJ(DEVICE1)  OBJTYPE(*DEV)  STATUS(*ON)
  RANGE(*NET)
```

This command varies on only the device. Note the RANGE parameter value has no effect on devices.

Example 5: Varying on the Line and Resetting the IOP

```
VRYCFG  CFGOBJ(LINE1)  OBJTYPE(*LIN)  STATUS(*ON)
  RANGE(*OBJ)  RESET(*YES)
```

This command varies on only the line and resets the associated IOP.

Example 6: Using Line Description Value for Wait Time

```
VRYCFG  CFGOBJ(LINE1)  OJBTYPE(*LIN)  STATUS(*ON)
  RANGE(*OBJ)  VRYWAIT(*CFGOBJ)
```

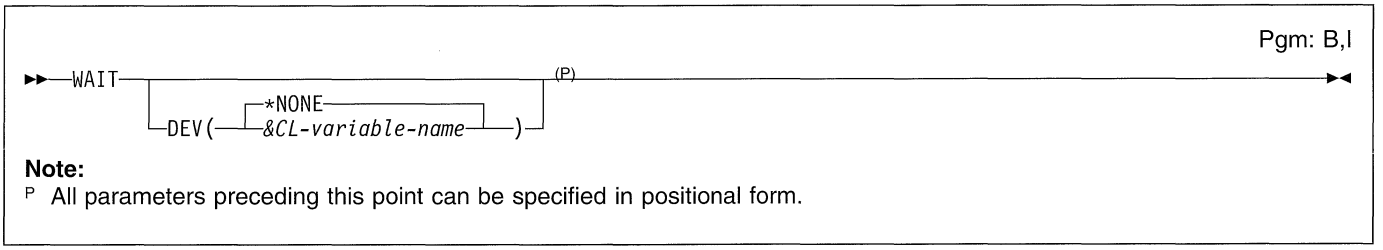
This command varies on only the line and uses the vary wait time value specified in the line description for LINE1.

Example 7: Using 80 Seconds as Vary Wait Time

```
VRYCFG  CFGOBJ(LINE1)  OBJTYPE(*LIN)  STATUS(*ON)
  RANGE(*OBJ)  VRYWAIT(80)
```

This command varies on only the line using 80 seconds as the vary wait time value.

WAIT (Wait) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Wait (WAIT) command accepts input from any display device from which user data is requested by one or more previous Receive File (RCVF), Send File (SNDF), or Send/Receive File (SNDRCVF) commands that do not wait to receive the input data. Those commands had *NO specified in the WAIT parameter or, in the case of SNDF, had the INVITE DDS keyword option specified in the record format sent to the display, and specified a particular device file to receive and transfer the data to the CL program. Only one input request per device can be outstanding at any given time. If there are multiple outstanding input requests, the user data of the first device to respond to the specified device file is sent to the CL program. If the data is received within the wait interval, the Wait operation ends and the next command in the program is processed. Otherwise an escape message is sent to the CL program.

The program waits the number of seconds specified on the WAITRCD keyword of the Create Display File (CRTDSPF), Change Display File (CHGDSPF), or Override with Display File (OVRDSPF) commands for a device to respond to an input request.

Restriction: This command is valid only for display files within a CL program. It cannot be used with database files.

Optional Parameters

DEV

Specifies the name of the CL variable that receives the name of the display device that responds with user data for the CL program.

***NONE:** No CL variable name is specified; the name of the responding device is not needed.

CL-variable-name: Specify the name (up to 10 characters) of the CL variable that receives the name of the responding device. A device name cannot be specified in this parameter.

Examples

Example 1: Receiving User Data

```
DCLF   FILE(MSCREEN)
  •
  •
  •
RCVF   DEV(DEV1)  WAIT(*NO)
  •
  •
  •
RCVF   DEV(DEV2)  WAIT(*NO)
  •
  •
  •
WAIT  DEV(&DEVNAM)
```

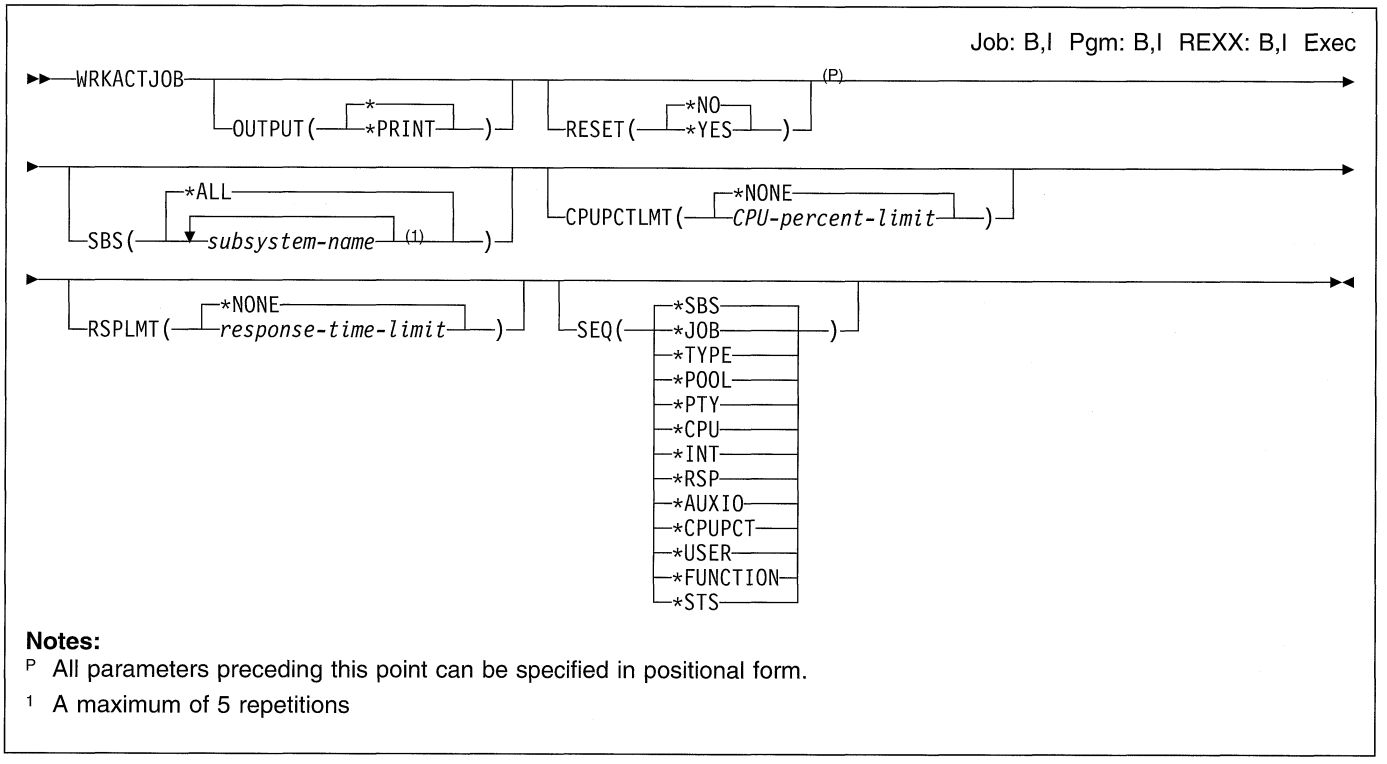
In this example, the device file MSCREEN is used to receive user data. The RCVF commands specify that the program does not wait for the data. The WAIT command causes the program to wait for the display device file MSCREEN to pass input data to it from one of its devices. The name of the responding display device is placed in the CL variable &DEVNAM. The received data is placed in the program variables associated with the record format of the declared file.

Example 2: Receiving Data in the Program Variable

```
DCLF   FILE(DF1)
  •
  •
  •
RCVF   DEV(DEV1)  WAIT(*NO)
  •
  •
  •
RCVF   DEV(DEV2)  WAIT(*NO)
  •
  •
  •
WAIT   DEV(&DN)
```

In this example, the RCVF commands specify that the program does not wait for user data. When the WAIT command is issued, the name of the responding device is placed in the CL variable &DN. The data received from the device file DF1 is placed in the program variable of the associated record format in the file DF1.

WRKACTJOB (Work with Active Jobs) Command



Notes:

| P All parameters preceding this point can be specified in positional form.

1 A maximum of 5 repetitions

Purpose

The Work with Active Jobs (WRKACTJOB) command allows the user to work with performance and status information for the active jobs in the system. The SEQ criteria can be changed by operations on the display. Selection values (CPUPCTLMT, SBS, RSPLMT) cannot be changed by operations on the display.

Optional Parameters

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

RESET

Specifies whether the active job statistics are reset.

***NO:** The active job statistics are not reset. The measurement time interval is extended (similar to pressing the F5 key) if a previous display active jobs command has run in the current job. All active jobs are displayed.

***YES:** The active job statistics are reset. A measurement time interval of zero (0) is used (similar to pressing the F13 key). All active jobs are displayed.

SBS

Specifies the names of the subsystems (or all subsystems) whose active jobs are displayed.

***ALL:** The active jobs in all subsystems are displayed. System jobs that are not associated with any subsystem are also displayed.

subsystem-name: Specify the name of the subsystem to be displayed. All active jobs in this subsystem (including the monitor) are displayed. Up to five subsystem names can be specified.

CPUPCTLMT

Specifies the minimum processing time percent value that a job must have before it is included on the display.

***NONE:** There is no minimum processing time limit that a job must have to be displayed.

CPU-percent-limit: Specify the minimum processing time percent limit that a job must have to be included on the display. Valid values range from 0.1 through 99.9.

RSPLMT

Specifies the minimum response time limit that a job must have before it is included on the display.

***NONE:** There is no minimum response time limit that a job must have to be displayed.

response-time-limit: Specify the minimum response time limit that a job must have to be included on the display. Valid values range from 0.1 through 999.9.

SEQ

Specifies the sequence of the active jobs that are shown.

***SBS:** The jobs are ordered on the basis of the subsystem in which they are running. Jobs that run in a subsystem (auto-start jobs, interactive jobs, batch jobs, readers, and writers) are put in alphabetical order by job name, and are indented under the subsystem with which they are associated. Subsystem monitors (with the jobs in the subsystem grouped under each monitor job) are put in alphabetical order and presented before system (SYS) jobs. The system jobs (start operating system, system arbiter, and LU services) are put in alphabetical order by job name, and are presented after the subsystem monitors and jobs in the subsystems.

***JOB:** Jobs are put in alphabetical order by job name.

***TYPE:** Jobs are put in alphabetical order by job type and job name within the same type.

***POOL:** Jobs are ordered by the system pool in which they are running. The lowest values are presented first.

***PTY:** Jobs are ordered by priority of running. The highest priority values (0) are presented first.

***CPU:** Jobs are ordered by the amount of processing time they have used since the job started. The largest values are presented first.

***INT:** Jobs are ordered by the number of operator interactions that have occurred during the measurement interval. The largest values are presented first. Jobs that have no interactions (blank interaction field) come after jobs with 0 interactions.

***RSP:** Jobs are ordered by the average response time during the measurement interval. The largest values are presented first. Jobs that have no interactions (blank interaction field) come after jobs with 0 response time.

***AUXIO:** Jobs are ordered by the number of auxiliary storage input/output (I/O) operations that have occurred during the measurement time interval. The largest values are presented first.

***CPUPCT:** Jobs are ordered by the percent of CPU resource they have used during the measurement interval. The largest values are presented first.

***USER:** Jobs are ordered alphabetically by user name.

***FUNCTION:** Jobs are put in alphabetical order by the contents of the function field.

***STS:** Jobs are put in alphabetical order by the contents of the status field.

Examples

Example 1: Resetting Active Job Statistics

```
WRKACTJOB  RESET(*YES)  CPUPCTLMT(2)
```

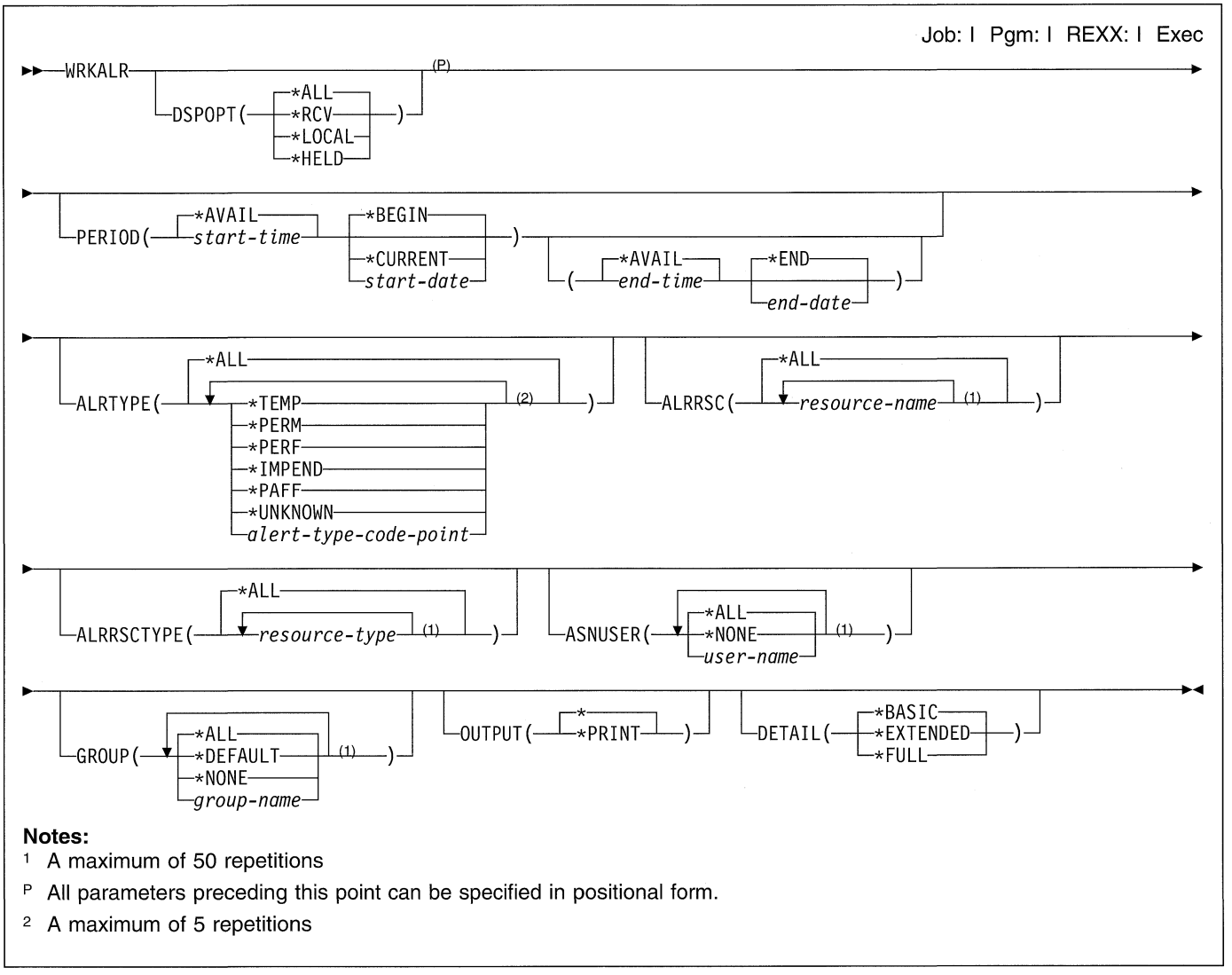
This command allows the user to work with a display with no jobs appearing; the active job statistics are reset and no job has used any processing unit time since the reset point. When the display appears, the F5 key may be pressed; this causes a display of all jobs that have exceeded 2 percent of the processing unit utilization since the reset point.

Example 2: Working With Jobs in a Subsystem

```
WRKACTJOB  SBS(QINTER)  SEQ(*INT)
```

This command allows the user to work with all jobs in the QINTER subsystem. The sequence of the jobs is by the number of operator interactions, with the job with the most interactions appearing first.

WRKALR (Work with Alerts) Command



Notes:

1 A maximum of 50 repetitions

P All parameters preceding this point can be specified in positional form.

2 A maximum of 5 repetitions

Purpose

The Work with Alerts (WRKALR) command allows the user to work with alerts that were created by the user's system or received from another system.

More information on alerts is in the *Alerts and DSNX Guide*.

Optional Parameters

DSPOPT

Specifies whether alerts received from other systems and/or created locally are shown. Alerts that cannot be sent to the system focal point and are marked held can be shown.

***ALL:** All alerts that are received and locally created are shown.

***RCV:** Only alerts received from other systems are shown.

***LOCAL:** Only locally created alerts are shown.

***HELD:** All alerts that cannot be sent to the system's focal point and are marked HELD are shown.

Note: There is a distinction between held alerts that are sent or forwarded by this system, and held alerts that are received by another system. DSPOPT(*HELD) shows only held alerts that could not be sent or forwarded by this system.

PERIOD

Specifies the period of time for which the logged data is shown. The following values can be coded in this parameter, which contains two lists of two elements each. If PERIOD is not specified, the following values are assumed:

```
PERIOD((*AVAIL *BEGIN)
    (*AVAIL *END))
```

Element 1: Starting Time

One of the following is used to specify the time at or after which the data must have been logged to be shown. Any entries logged before the specified time and date are not shown.

***AVAIL:** The logged data that is available for the specified starting *date* is shown.

start-time: Specify the starting time for the specified starting date that indicates the logged data that is shown. The time is specified in 24-hour format with or without a time separator as follows:

- With a time separator, specify a string of 5 or 8 digits where the time separator separates the hours, minutes, and seconds. If this command is entered from the command line, the string must be enclosed in apostrophes. If a time separator other than the separator specified for your job is used, this command fails.
- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where **hh** = hours, **mm** = minutes, and **ss** = seconds. Valid values for **hh** range from 00 through 23. Valid values for **mm** and **ss** range from 00 through 59.

Element 2: Starting Date

One of the following is used to specify the starting date on which or after which the data must have been logged. Any entries logged before the specified date are not shown.

***BEGIN:** The logged data from the beginning of the log is shown.

Note: If *BEGIN is specified, then any time value other than *AVAIL for start-time is ignored.

***CURRENT:** The logged data for the current day and between the specified starting and ending times (if specified) is shown.

start-date: Specify the date for which logged data is shown. The date must be entered in the format specified by the system values QDATFMT and, if separators are used, QDATSEP.

Element 3: Ending Time

One of the following is used to specify the ending time before which the data must have been logged.

***AVAIL:** The logged data that is available for the specified ending *date* is shown.

end-time: Specify the ending time for the specified ending date that determines the logged data that is shown. See the description of *start-time* for details about how time can be specified.

Element 4: Ending Date

One of the following is used to specify the ending date before which or on which the data must have been logged.

***END:** The last day on which data was logged is shown. If PERIOD(*END) is specified, a time value other than *AVAIL for end-time is ignored.

end-date: Specify the ending date for which logged data is to be shown. The date must be entered in the format specified by the system values QDATFMT and, if separators are used, QDATSEP.

ALRTYPE

Specifies which types of alerts are shown. The alert type indicates the severity of the alert.

***ALL:** All types of alerts are shown.

Up to five of the following types of alerts can be specified.

***TEMP:** Alerts reporting a temporary problem are shown.

***PERM:** Alerts reporting a permanent problem are shown.

***PERF:** Alerts reporting a performance problem are shown.

***IMPEND:** Alerts reporting an impending problem are shown.

***PAFF:** Alerts reporting a problem about a permanently affected resource are shown. These alerts indicate that their originator has determined that the target resource is lost because of a persistent error in a resource other than the target.

***UNKNOWN:** Alerts reporting a problem with unknown severity are shown.

alert-type-code-point: Specify the code point for the alert type. The code point is specified by two (2) hexadecimal digits.

ALRRSC

Specifies the resource name of alerts that are reporting problems. Up to 50 resource names can be specified.

***ALL:** Alerts about failing resources are shown.

resource-name: Alerts that are reporting problems associated with the assigned resource name are shown.

ALRRSCTYPE

Specifies the resource types of alerts that are reporting problems. Up to 50 resource types can be specified. Each resource name has a resource type associated with that resource. Resource types are diskette (DKT) or tape (TAP), for example.

***ALL:** Alerts for all resource types are shown.

resource-type: Alerts that are reporting problems associated with the assigned resource type are shown.

ASNUSER

Specifies the user to which the alerts being shown are assigned. This value is taken from the value on the ASNUSER parameter in the Add Alert Action Entry (ADDALRACNE) command.

***ALL:** All alerts are shown.

***NONE:** The alerts not assigned to a user are shown.

user-name: Specify the name of the user to which the alerts being shown are assigned.

GROUP

Specifies the group to which the alerts being shown are assigned. This value is taken from the value on the GROUP parameter in the Add Alert Selection Entry (ADDALRSLTE) command.

***ALL:** All alerts are shown.

***DEFAULT:** The alerts assigned to the default group are shown.

***NONE:** The alerts not assigned to a group are shown.

group-name: Specify the name of the group to which the alerts being shown are assigned.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

DETAIL

Specifies the level of detail contained in the printed listing. This parameter is valid only if OUTPUT(*PRINT) is specified.

***BASIC:** The output is a list of the alert resource, alert type, date, time, problem ID, assigned user, group assigned, alert description, and probable cause. The information on this list is only for alerts that meet requirements set by the subsetting parameters.

***EXTENDED:** The output is a spool file containing the WRKALR screen information, alert hierarchy, probable cause and recommended action, and details from the main detail panel. The spool file contains information only for the alerts that meet the user criteria.

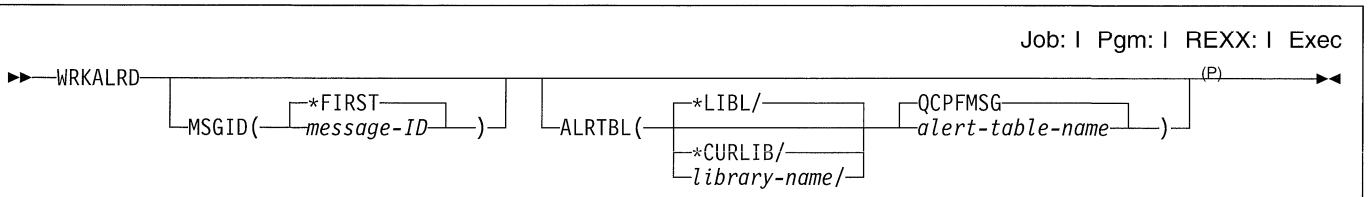
***FULL:** The output is a spool file containing the same data as the *EXTENDED value output, plus additional information from the detail menu screens. Only the panels that contain data are printed.

Example

```
WRKALR  DSPORT(*LOCAL)  ALRTYPE(*TEMP *PERM)
  ALRRSCTYPE(DKT)
```

This command allows the user to work with all locally created alerts in the alert database that are both temporary and permanent. The alerts shown are reporting problems about diskettes.

WRKALRD (Work with Alert Descriptions) Command



Note:

(P) All parameters preceding this point can be specified in positional form.

Purpose

The Work with Alert Descriptions (WRKALRD) command allows the user to show, add, change, and remove alert description records. More information on alerts is in the *Alerts and DSNX Guide*.

Optional Parameters

MSGID

Specifies the message ID at which the Work with Alert Description display starts.

***FIRST:** The display begins with the first message identifier description found in the alert table.

message-ID: Specify the message identifier.

ALRTBL

Specifies the qualified name of the alert table.

The name of the alert table can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

QCPFMSG: The alert table named QCPFMSG is used.

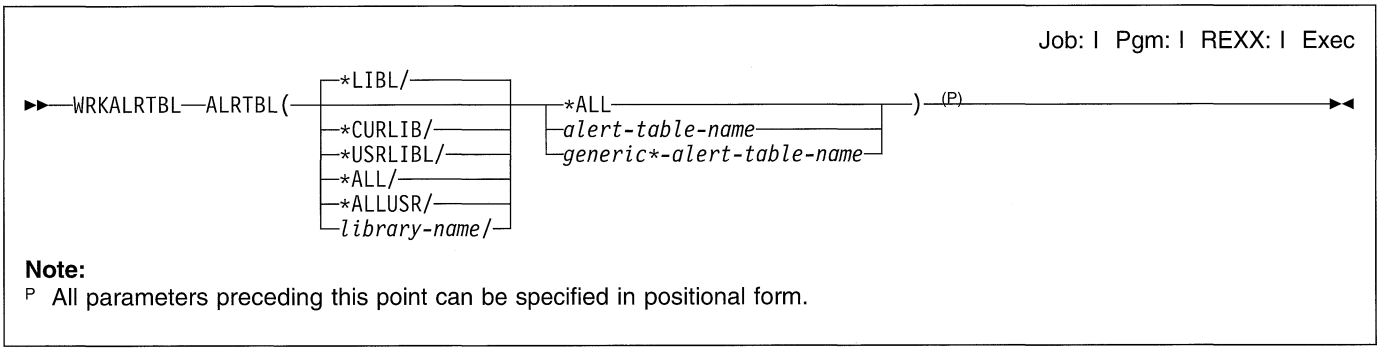
alert-table-name: Specify the name of the alert table.

Example

```
WRKALRD  MSGID(USR1234)  ALRTBL(USER/USRMSGS)
```

This command shows the Work with Alert Description display, starting with message identifier USR1234 from alert table USRMSGS in library USER.

WRKALRTBL (Work with Alert Tables) Command



Note:
| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Alert Table (WRKALRTBL) command allows you to display and work with a list of alert tables, to change and delete specified alert tables, to work with the alert descriptions contained in specified alert tables, and to create new alert tables. More information on the alerts is in the *Alerts and DSNX Guide*.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the alert tables to which you have some authority will be shown on the display.
3. To perform operations on the alert tables, you must have USE authority to the command used by the operation, and the appropriate authority to the alert tables on which the operation is to be performed.

Required Parameters

ALRTBL

Specifies the qualified name of the alert table list that is shown.

| The name of the alert table can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All the alert tables specified in the library are listed.

alert-table-name: Specify the name of the alert table that is shown.

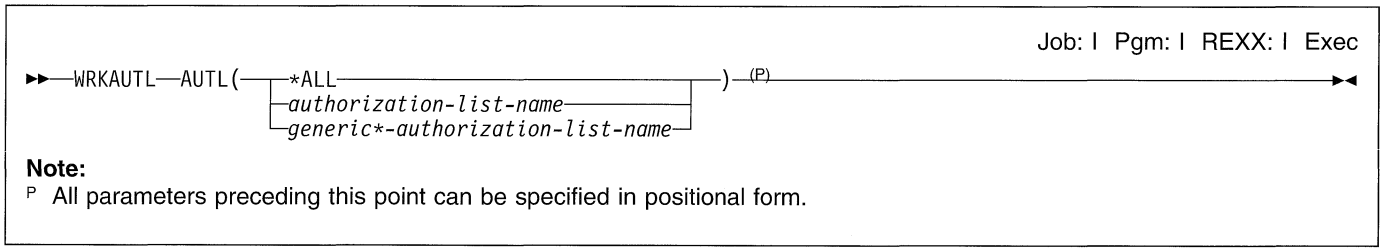
generic-alert-table-name:* Specify the generic name of the alert table. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Example

```
WRKALRTBL  ALRTBL(ALRTBLLIB/AL*)
```

This command shows a list of all alert tables in library ALRTBLLIB whose names begin with AL. From the list shown, you can change, delete, or work with the alert descriptions in any or all of the alert tables shown. You can also create a new alert table.

WRKAUTL (Work with Authorization Lists) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Authorization Lists (WRKAUTL) command allows you to work with authorization lists. With this command, you can display, edit, delete, display the list's objects, or change the text for an authorization list.

Restrictions:

1. Only the authorization lists to which you have some authority will be shown on the display.
2. To perform operations on the authorization lists, you must have USE authority to the command used by the operation, and the appropriate authority to the authorization list on which the operation is to be performed.

Required Parameters

AUTL

Specifies the name of an authorization list with which you want to work.

***ALL:** A list of all the authorization lists that you either own or have authority to see is shown.

authorization-list-name: Specify the authorization list with which you want to work.

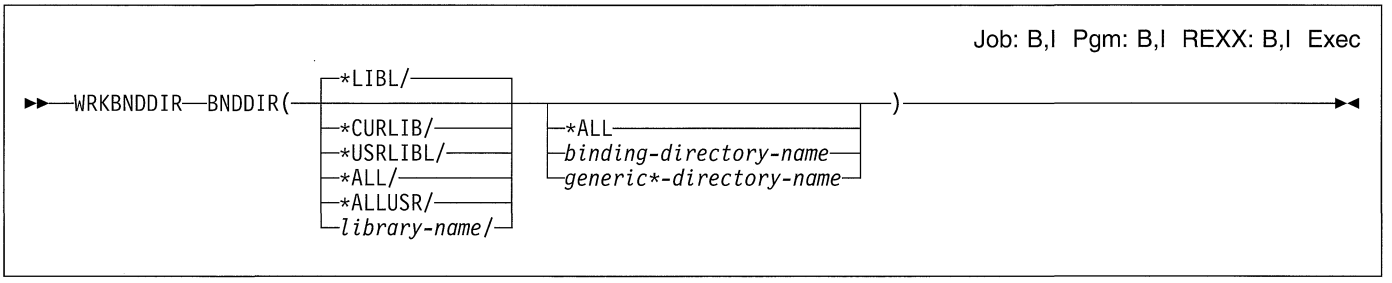
generic-authorization-list-name:* Specify the generic name of the authorization list. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Example

```
WRKAUTL  AUTL(FR*)
```

This command allows you to work with a list of all the authorization lists that begin with FR that you have authority to see.

WRKBNDDIR (Work with Binding Directories) Command



Purpose

The Work with Binding Directory (WRKBNDDIR) command allows you to display and work with a list of binding directories.

Restrictions:

1. Only the libraries to which you have *USE authority are searched.
2. Only the binding directories to which you have some authority are shown on the display.
3. To perform operations on the binding directories, you must have *USE authority to the command and the appropriate authority to the binding directory on which the operation is performed.

Required Parameters

BNDDIR

Specifies the binding directory to work with.

The name of the binding directories can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** Find all binding directories in the specified library or libraries.

binding-directory-name: Specify the name of the binding directory to work with.

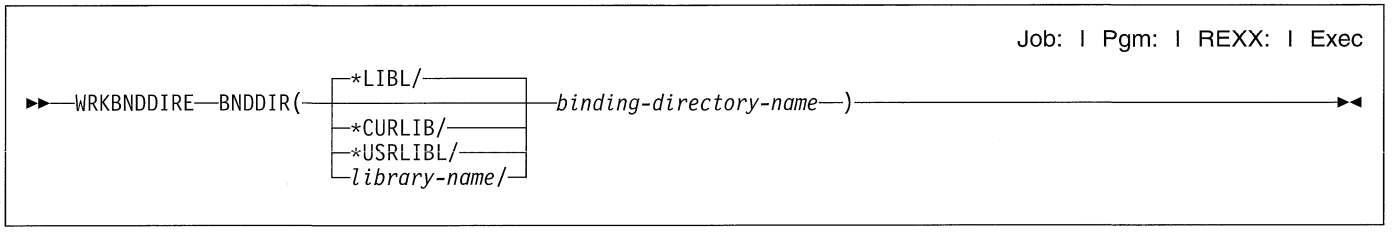
generic-directory-name:* Specify the generic name of the binding directories. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all service programs with names that begin with the generic name, and for which the user has authority, are shown. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete service program name.

Example

```
WRKBNDDIR   BNDDIR(HOLDER)
```

This command allows you to work with a binding directory named HOLDER.

WRKBNDDIRE (Work with Binding Directory Entries) Command



Purpose

The Work with Binding Directory Entries (WRKBNDDIRE) command allows you to work with the entries in a binding directory.

Restrictions:

1. Only the libraries to which you have *USE authority are searched.
2. To perform operations on the binding directories, you must have *USE authority to the command and the appropriate authority to the binding directory on which the operation is to be performed.

Required Parameters

BNDDIR

Work with the entries in the specified binding directory.

The name of the binding directory can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

library-name: Specify the name of the library to be searched.

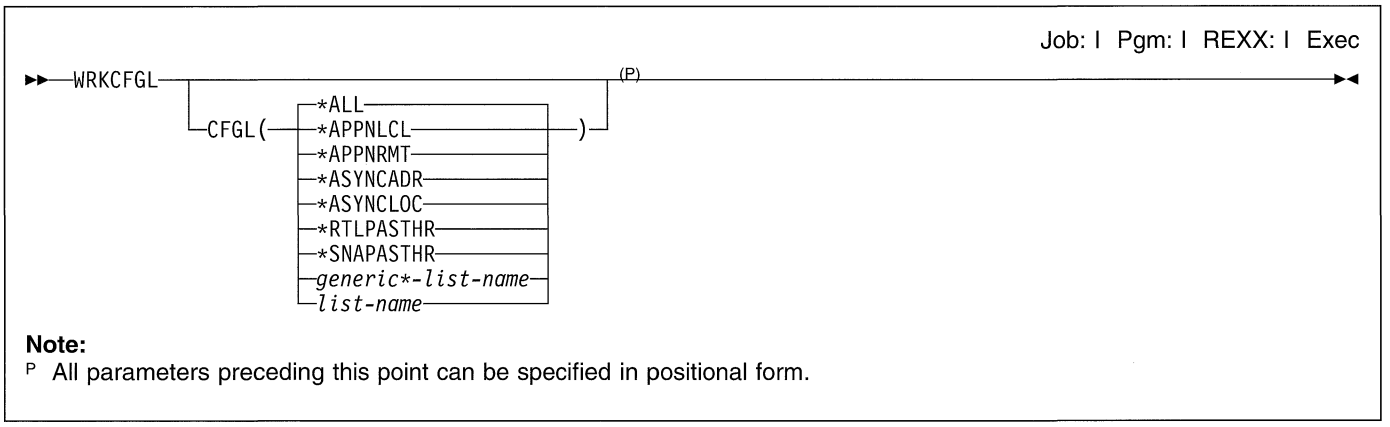
binding-directory-name: Specify the name of the binding directory whose entries are to be shown.

Example

```
WRKBNDDIRE   BNDDIR(COINS)
```

This command allows you to work with the entries in binding directory COINS.

WRKCFGL (Work with Configuration Lists) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Configuration Lists (WRKCFGL) command allows you to show and work with configuration list functions by using the Work with Configuration Lists display. This command shows the Work with Configuration Lists display.

Optional Parameters

CFGL

Specifies the configuration lists with which to work.

***ALL:** Work with all configuration lists.

***APPNLCL:** Work with the advanced peer-to-peer networking (APPN) local location configuration list.

***APPNRMT:** Work with the APPN remote location configuration list.

***ASYNCADR:** Work with asynchronous packet assembly/disassembly (PAD) network address configuration lists.

***ASYNCLOC:** Work with the asynchronous remote location configuration list.

***RTLPASTHR:** Work with the retail pass-through configuration list.

***SNAPASTHR:** Work with the SNA pass-through configuration list.

generic-list-name:* Specify the generic name of the configuration list. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

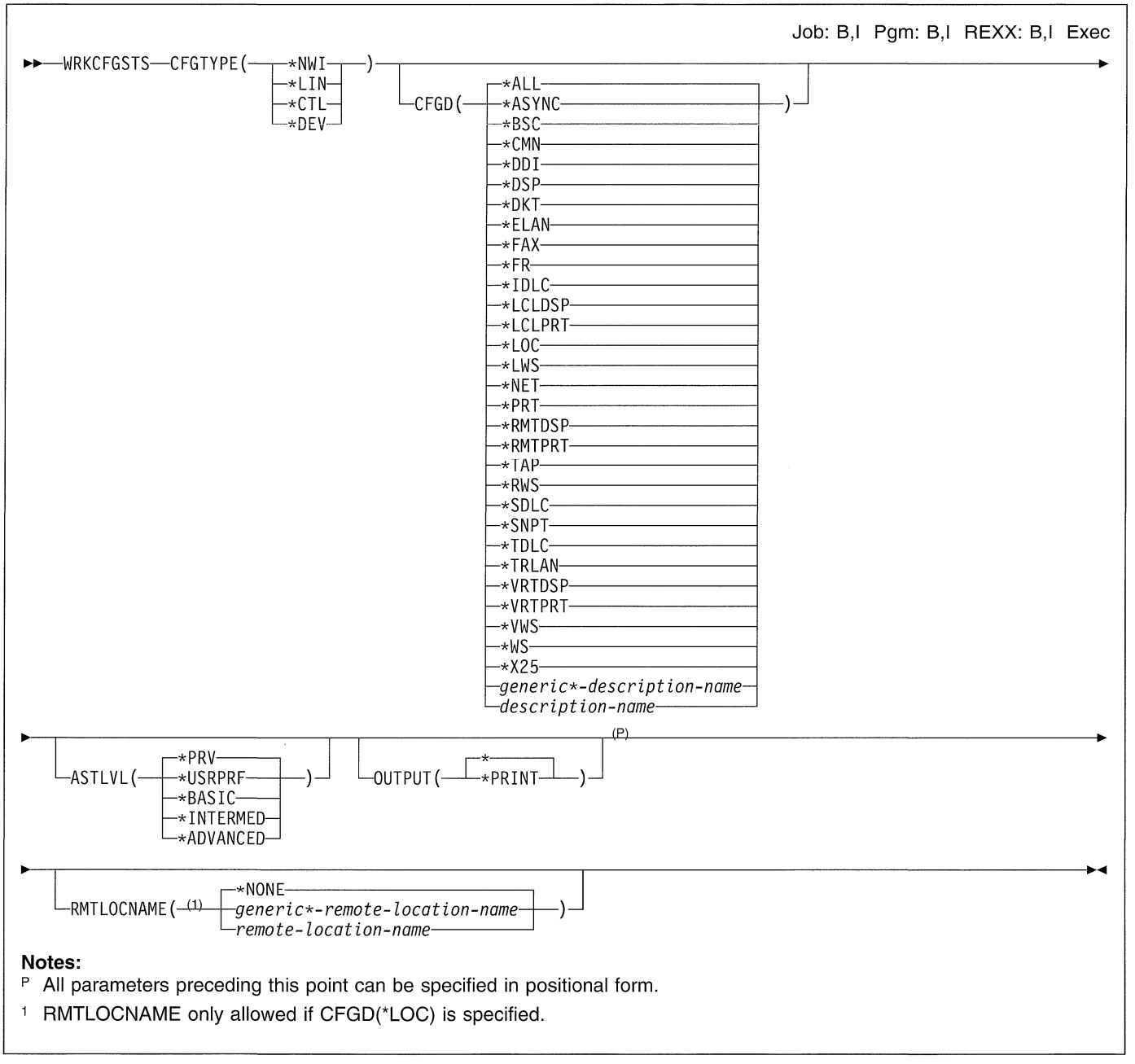
list-name: Specify a single configuration list name.

Example

```
WRKCFGL  CFGL(PEG*)
```

This command allows you to utilize the Work with Configuration Lists display to work with entries for all configuration lists whose names start with the PEG prefix.

WRKCFGSTS (Work with Configuration Status) Command



Purpose

The Work with Configuration Status (WRKCFGSTS) command allows you to display and work with configuration status functions. When this command is entered, the Work with Configuration Status display is shown. This display shows status information for network interfaces, lines, controllers, and/or devices, and for jobs associated with devices. The display can be for a location or for one or more network interfaces, lines, controllers, or devices. All attached configuration descriptions are shown for each network interface, line, controller, or device description selected. The names of the attached configuration descriptions are indented under the name of the object to which they are attached. If status is requested for a specific network interface, line, controller, or device, both upline and downline attachments are shown. If status is requested for any collection of controllers or devices, only downline attachments are shown. Status for a location shows devices and modes for the specified location.

Options available on the Work with Configuration Status display are to vary status, end or resume recovery, hold or release a device, work with a job, or display location path. Options can be entered for more than one configuration object at a time. Option requests are placed in a list and processed sequentially.

Required Parameters

CFGTYPE

Specifies the type of description for which to show status.

***NWI:** Status for network interfaces is shown.

***LIN:** Status for lines is shown.

***CTL:** Status for controllers is shown.

***DEV:** Status for devices is shown.

Optional Parameters

CFGD

Specifies the descriptions being placed on the Work with Configuration Status display.

***ALL:** All descriptions of the type specified for the CFGTYPE parameter are shown.

***ASYNC:** Status descriptions of asynchronous lines are shown.

***BSC:** Status descriptions of bisynchronous lines are shown.

***CMN:** Status descriptions of communications controllers and/or devices are shown.

***DDI:** Status descriptions of the distributed data interface lines are shown.

***DSP:** Status descriptions of both local and remote display station devices are shown.

***DKT:** Status descriptions of diskette devices are shown.

***ELAN:** Status descriptions of Ethernet local area network (LAN) lines are shown.

***FAX:** Status for all facimile (fax) lines is displayed.

***FR:** Status descriptions of the frame relay lines are shown.

***IDLC:** Status descriptions of ISDN data link control (IDLC) lines are shown.

***LCLDSP:** Status descriptions of local display station devices are shown.

***LCLPRT:** Status descriptions of local printer devices are shown.

***LOC:** Status descriptions of communications devices whose remote location name matches the name specified for the RMTLOCNAME parameter are shown. A value, other than *NONE, must be specified for the RMTLOCNAME parameter if you specify *LOC for the CFGD parameter.

***LWS:** Status descriptions of local work station controllers are shown.

***NET:** Status descriptions of network lines are shown.

***PRT:** Status descriptions of both local and remote printer devices are shown.

***RMTDSP:** Status descriptions of remote display station devices are shown.

***RMTPRT:** Status descriptions of remote printer devices are shown.

***TAP:** Status descriptions of tape devices are shown.

***RWS:** Status descriptions of remote work station controllers are shown.

***SDLC:** Status descriptions of synchronous data link control (SDLC) lines are shown.

***SNPT:** Status descriptions of SNA pass-through devices are shown.

***TDLC:** Status descriptions of twinaxial data link control (TDLC) lines are shown.

***TRLAN:** Status descriptions of token ring lines are shown.

***VRTDSP:** Status descriptions of virtual display station pass-through devices are shown.

***VRTPRT:** Status descriptions of virtual (pass-through) printer devices are shown.

***VWS:** Status descriptions of virtual work station pass-through controllers are shown.

***WS:** Status descriptions of work station controllers are shown.

***X25:** Status descriptions of X.25 lines are shown.

generic-description-name:* Specify the generic name of the description. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

description-name: Specify display of a specific description and any upline attachments as well as downline attachments.

ASTLVL

Specifies the user interface to use.

***PRV:** The previous user interface is used.

***USRPRF:** The user interface specified in the user profile is used.

***BASIC:** The Operational Assistant user interface is used.

***INTERMED:** The system interface is used.

***ADVANCED:** The expert system interface is used.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

RMTLOCNAME

Specifies the remote location name of communications devices for which status is shown. Only devices with the specified remote location name are listed on the Work with Configuration Status display.

This parameter is required if CFGD(*LOC) is specified. It is not a valid parameter for any other value of CFGD.

***NONE:** The descriptions shown are not for communications devices with a specific location name. *NONE must be specified if a value other than *LOC is specified on the CFGD parameter.

generic-remote-location-name:* Specify the generic name of the remote location.

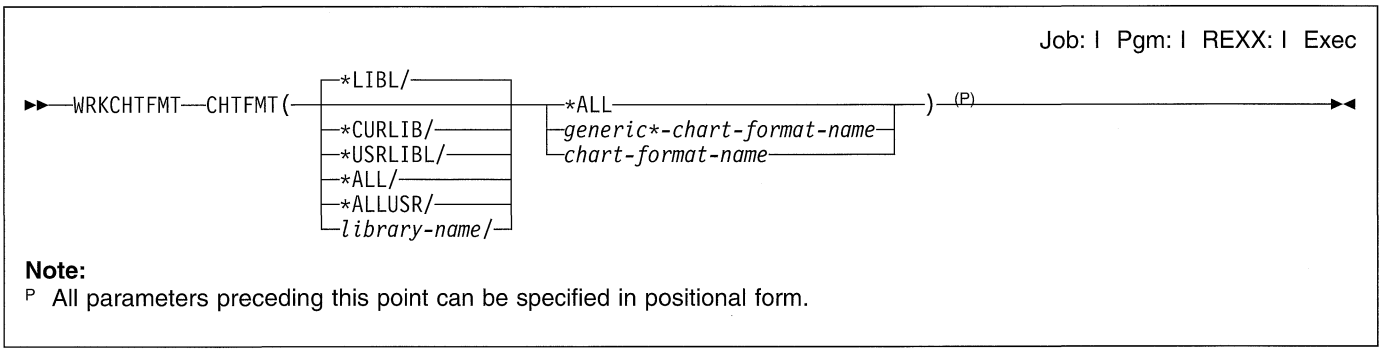
remote-location-name: Specify the full name of a remote location.

Example

```
WRKCFGSTS  CFGTYPE(*DEV)  CFGD(*RMTDSP)
```

This command allows the user to utilize the Work with Configuration Status command to show the status for all remote display stations.

WRKCHTFMT (Work with Chart Formats) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Chart Formats (WRKCHTFMT) command allows the user to display and work with a list of chart formats from one or more libraries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the chart formats to which you have some authority will be shown on the display.
3. To perform operations on the chart formats, you must have USE authority to the command used by the operation, and appropriate authority to the chart formats on which the operation is to be performed.

Required Parameters

CHTFMT

Specifies a list of chart formats in the libraries that are shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the chart formats.

The name of the chart format can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All chart formats in the libraries identified in the library qualifier are shown.

generic-chart-format-name:* Specify the generic name of the chart format. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

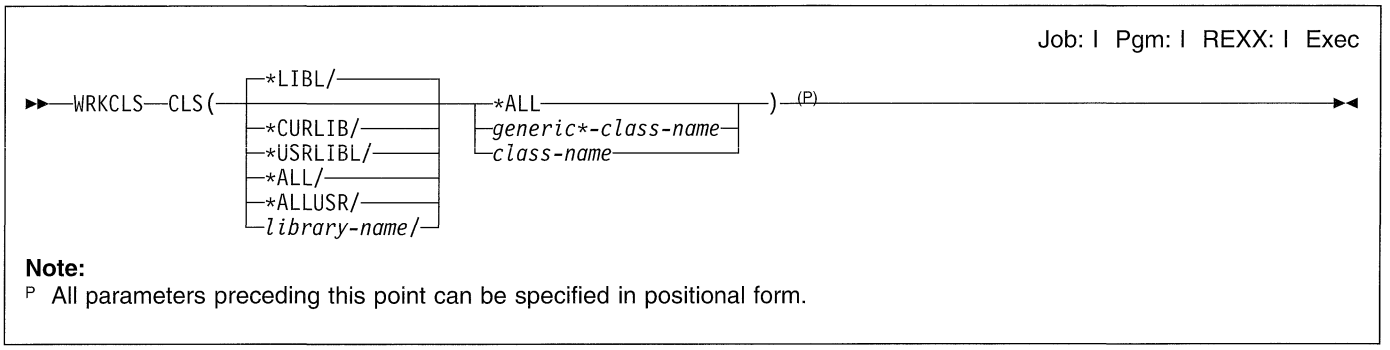
chart-format-name: Specify the name of the chart format that is shown.

Example

```
WRKCHTFMT  CHTFMT(LIB01/ABC*)
```

This command allows you to work with a list of chart formats *beginning* with ABC that are stored in library LIB01.

WRKCLS (Work with Classes) Command



Note:

| (P) All parameters preceding this point can be specified in positional form.

Purpose

The Work with Classes (WRKCLS) command allows the user to display and work with a list of classes from one or more libraries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the classes to which you have some authority will be shown on the display.
3. To perform operations on the classes, you must have USE authority to the command used by the operation, and the appropriate authority to the classes on which the operation is to be performed.

Required Parameters

CLS

Specifies a list of class descriptions in the library or libraries shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the class descriptions. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

The name of the class description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All class descriptions in the libraries identified in the library qualifier are shown (except those class descriptions for which the user does not have some authority).

generic-class-name:* Specify the generic name of the class. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

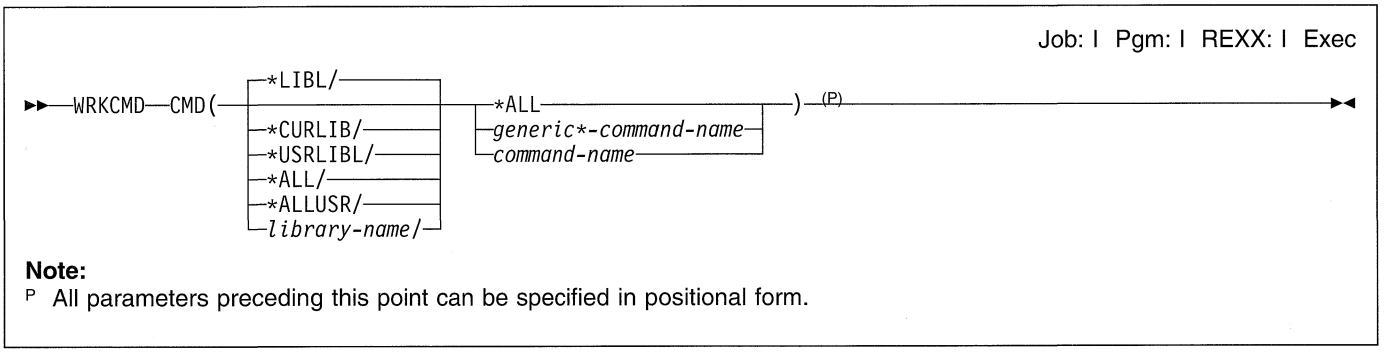
class-name: Specify the name of the class description that is shown.

Example

```
WRKCLS  CLS(LIB01/ABC*)
```

This command allows you to display and work with a list of classes beginning with ABC stored in library LIB01.

WRKCMD (Work with Commands) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Commands (WRKCMD) command allows you to display and work with a list of commands from a user-specified subset of command names. Several command-related functions are available from this list.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the commands to which you have some authority will be shown on the display.
3. To perform operations on the commands, you must have USE authority to the command used by the operation, and the appropriate authority to the commands on which the operation is to be performed.

Required Parameters

CMD

Specifies the qualified name of the command listed on the Work with Commands display. A specific command or a generic command can be specified. Optionally, either type can be qualified by a library name.

| The name of the command can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All commands in the specified libraries are listed on the Work with Commands display.

generic-command-name:* Specify the generic name of the command list. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

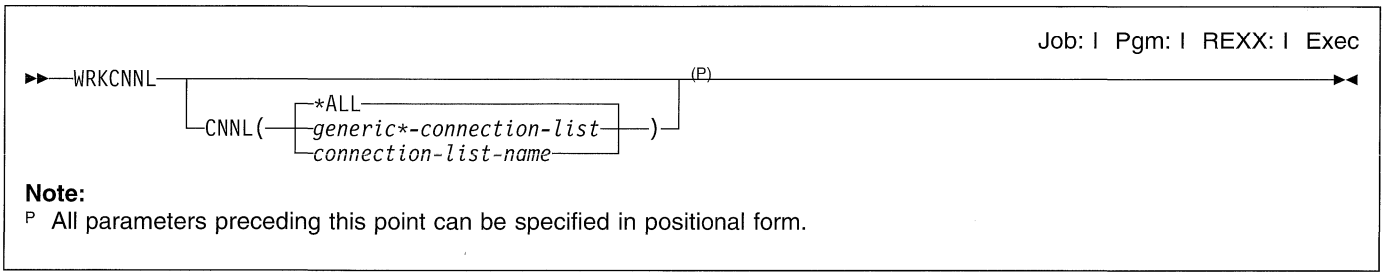
command-name: Specify the name of the command being listed.

Example

```
WRKCMD  CMD(QGPL/DSP*)
```

This command allows you to work with a list of all commands in the QGPL library that start with DSP.

WRKCNNL (Work with Connection Lists) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Connection Lists (WRKCNNL) command allows the user to work with a connection list.

Optional Parameters

CNNL

Specifies the connection list to work with.

***ALL:** All connection lists are worked with.

generic-connection-list-name:* Specify the generic name of the connection list. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

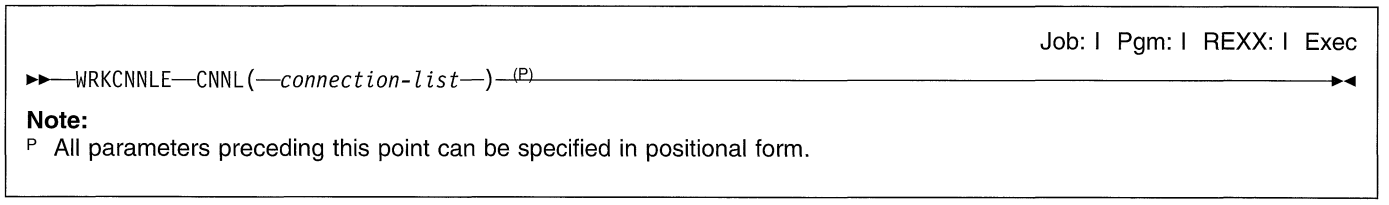
connection-list-name: Specify a specific connection list name.

Example

```
WRKCNNL  CNNL(CHI*)
```

This command allows the user to use the Work with Connection List display to work with connection lists which have names that begin with 'CHI' and for which the user has authority.

WRKCNNLE (Work with Connection List Entries) Command



Purpose

The Work with Connection List Entries (WRKCNNLE) command allows the user to work with specific connection list entries.

Required Parameters

CNNL

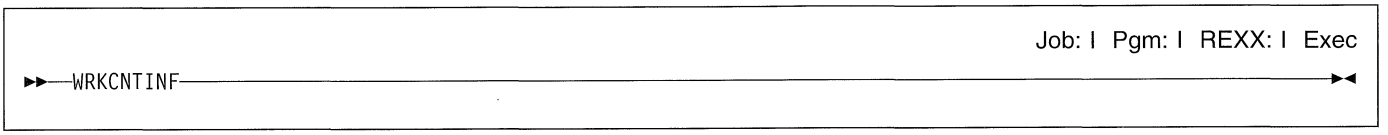
Specify the connection list containing the entries to work with.

Example

```
WRKCNNLE  CNNL(CHICAGO)
```

This command allows the user to use the Work with Connection List Entries display to work with the entries in connection list CHICAGO.

WRKCNTINF (Work with Contact Information) Command



Purpose

The Work with Contact Information (WRKCNTINF) command allows you to display and work with the information that helps you contact, or be contacted by, various support centers. When this command is specified, a display appears that allows selection of one of six kinds of support functions.

More information on this command is in the *Q & A Database Coordinator's Guide*.

Restrictions: 1) This command is shipped with public *EXCLUDE authority and the QSRV and QSRVBAS user profiles have private authorities to use the command. 2) This command is valid only in an interactive environment.

There are no parameters for this command.

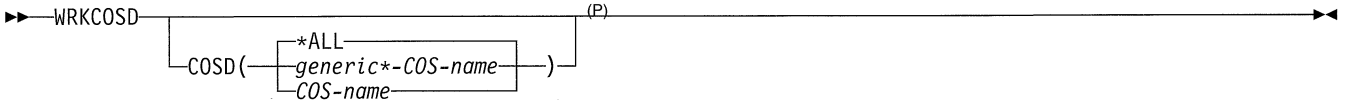
Example

```
WRKCNTINF
```

This command allows you to work with the Work with Support Contact Information display.

WRKCOSD (Work with Class-of-Service Descriptions) Command

Job: I Pgm: I REXX: I Exec



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

This command shows the Work with Class-of-Service Description display, which allows you to display and work with class-of-service description functions.

Optional Parameters

COSD

Specifies the class-of-service descriptions with which you want to work. which class-of-service descriptions to include in the list of class-of-service descriptions on the Work with Class-of-Service Descriptions display.

***ALL:** You can work with all class-of-service descriptions.

generic-COS-name:* Specify the generic name of the COS. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

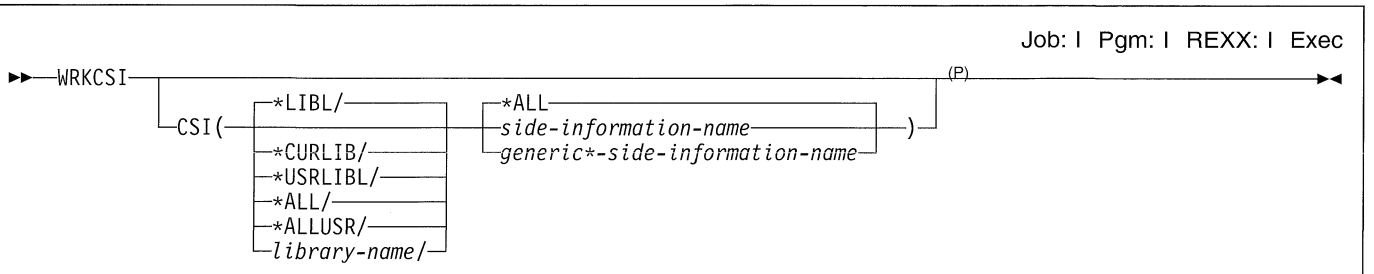
COS-name: Specify a single class-of-service description name.

Example

```
WRKCOSD  COSD(MPLS*)
```

This command allows you to utilize the Work with Class-of-Service Descriptions command to display entries for all class-of-service descriptions whose names start with MPLS.

WRKCSI (Work with Communications Side Information) Command



Note:
| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Communications Side Information (WRKCSI) command is used to work with side information objects, specified by the side information object name, from the library or libraries specified.

Optional Parameters

CSI

Specifies the name of the side information object to work with.

| The name of the side information object can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

side-information-name: Specify the name of the object that contains the desired side information. This is the Common Programming Interface (CPI)-Communications symbolic destination name (sym_dest_name). If *LIBL or *USRLIBL is specified as the library name, only the first side information object found with the specified name is listed.

generic-side-information-name:* Specify the generic
| name of the side information object. A generic name is
| a character string of one or more characters followed by
| an asterisk (*); for example, ABC*. The asterisk substi-
| tutes for any valid characters. A generic name specifies
| all objects with names that begin with the generic prefix
| for which the user has authority. If an asterisk is not
| included with the generic (prefix) name, the system
| assumes it to be the complete object name. If the com-
| plete object name is specified, and multiple libraries are
| searched, multiple objects can be worked with only if
| *ALL or *ALLUSR library values can be specified for the
| name. For more information on the use of generic func-
| tions, refer to "Rules for Specifying Names."

Examples

Example 1: Displaying Information Objects

```
WRKCSI
```

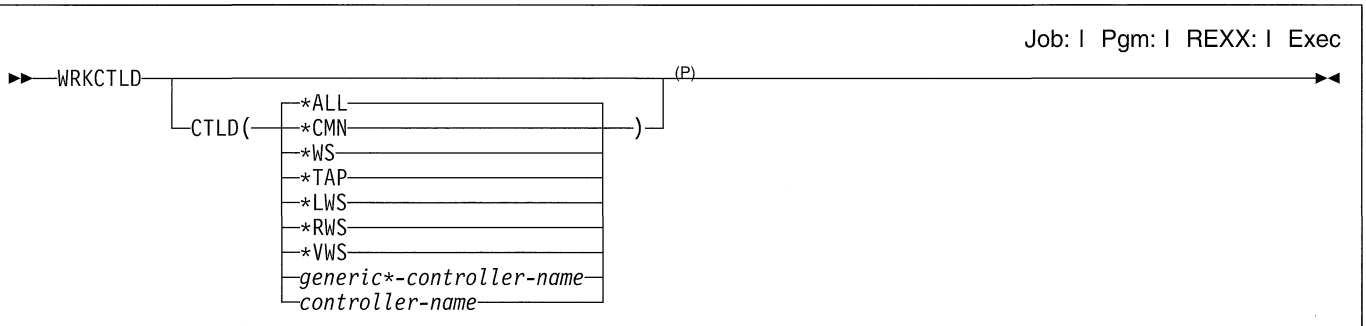
This command displays all the side information objects in the library list. The user can work with the objects.

Example 2: Displaying Objects the Begin with "SIDE"

```
WRKCSI  CSI(QGPL/SIDE*)
```

This command displays all the side information objects that begin with the characters "SIDE" and are located in library QGPL. The user can work with the objects.

WRKCTLD (Work with Controller Descriptions) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Controller Descriptions (WRKCTLD) command allows you to display and work with controller description functions by using the Work with Controller Descriptions display. This command shows the Work with Controller Descriptions display, which displays a list of controller descriptions to which you are authorized.

From the Work with Controller Descriptions display, you can use all of the available configuration functions with your controller descriptions. You can create, change, copy, delete, display, and print your controller descriptions.

This command also allows you to group the controller descriptions that appear on the Work with Controller Descriptions display. You can have all of the controller descriptions, only those controller descriptions of a certain type, or only those controller descriptions whose names start with a certain character string on the display.

Optional Parameters

CTLD

Specifies which controller descriptions to include in the list of controller descriptions on the Work with Controller Descriptions display.

***ALL:** The user works with all controller descriptions.

***CMN:** The user works with communications controller descriptions only.

***WS:** The user works with work station controller descriptions only.

***TAP:** The user works with tape controller descriptions only.

***LWS:** The user works with local work station controller descriptions only.

***RWS:** The user works with remote work station controller descriptions only.

***VWS:** The user works with virtual work station controller descriptions only.

generic-controller-name:* Specify the generic name of the controller. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

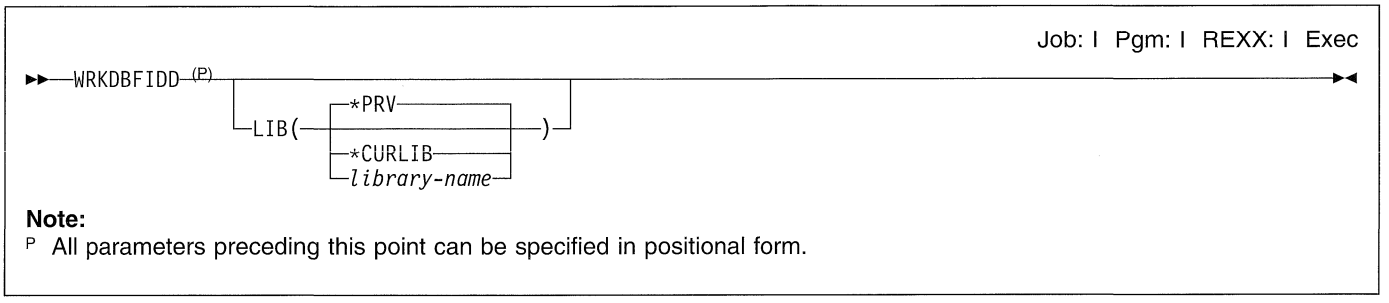
controller-name: Specify the name of a controller description.

Example

```
WRKCTLD  CTLD(*LWS)
```

This command allows you to use the Work with Controller Descriptions display to work with entries for all local work station controllers to which you have authority.

WRKDBFIDD (Work with Database Files Using IDDU) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Database Files Using the Interactive Data Definition Utility (IDDU) (WRKDBFIDD) command allows you to select an option from the IDDU Work with Database Files display to create physical files or enter data into a file.

Optional Parameters

LIB

Specifies the name of the library that contains the files.

The name of the file can be qualified by one of the following library values:

***PRV:** The files are located in the last library the user worked with in the IDDU. If this is the first time the user works with the IDDU, the current library is used.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

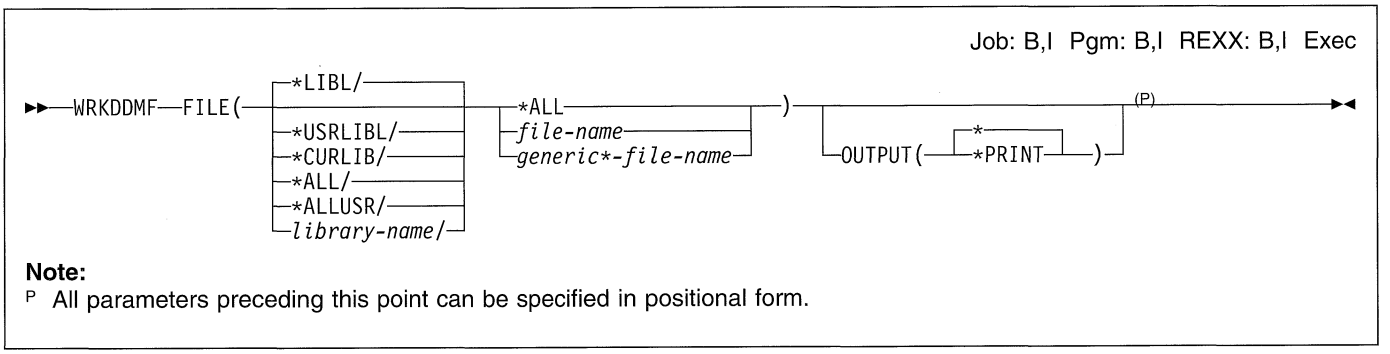
library-name: Specify the name of the library to be searched.

Example

```
WRKDBFIDD  DEPT245
```

This command allows you to work with database files in the DEPT245 library using the IDDU Work with Database Files display.

WRKDDMF (Work with Distributed Data Management Files) Command



Note:
(P) All parameters preceding this point can be specified in positional form.

Purpose

The Work with Distributed Data Management Files (WRKDDMF) command allows users to work with a list of DDM files. From this list, users can change, delete, or create DDM files. In addition, users can print or display the details of the command or print the list of DDM files.

Required Parameters

FILE

Specifies the qualified name of the DDM file whose description is being changed.

The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All files in the library or libraries specified by the library qualifier are to have their descriptions shown.

file-name: Specify the name of the file that has its description shown.

generic-file-name:* Specify the generic name of the file. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Optional Parameters

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output.

*: Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

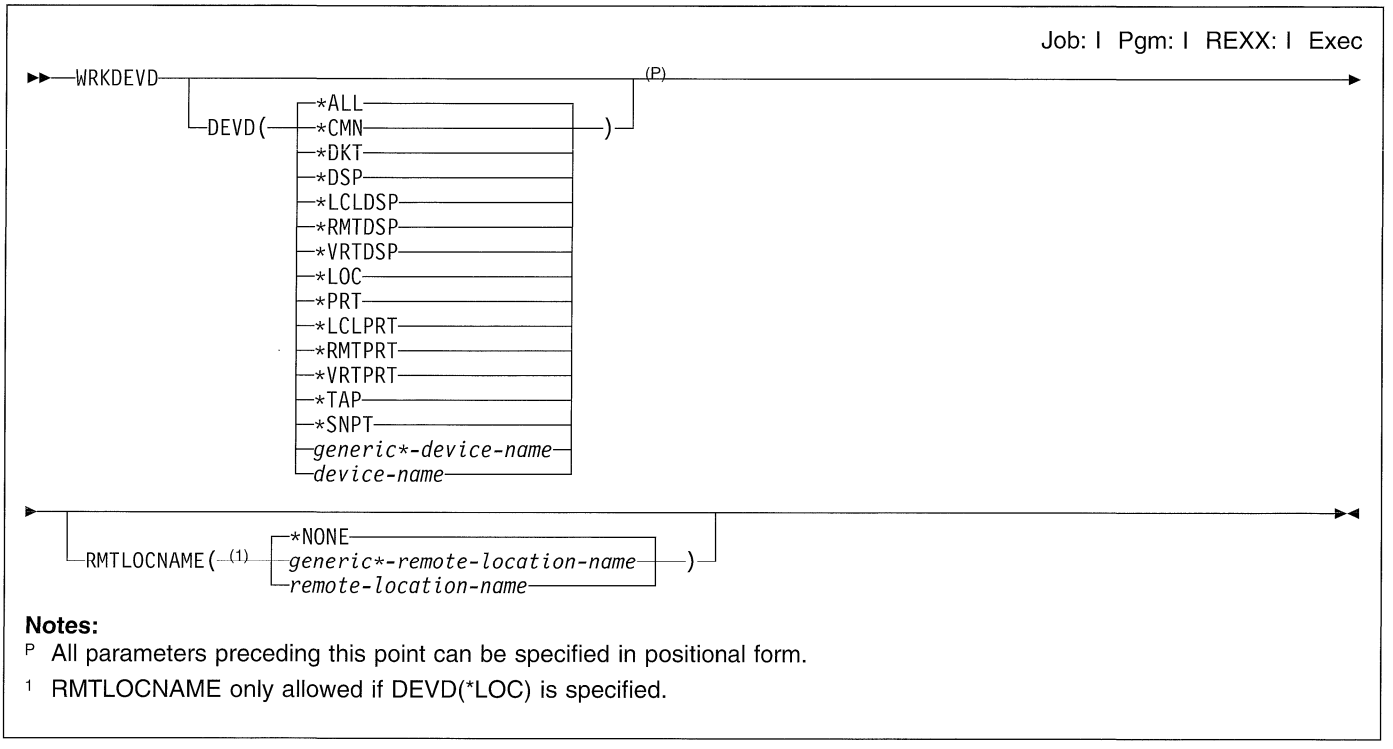
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKDDMF
```

This command allows the user to work with the Work with DDM Files display.

WRKDEVD (Work with Device Descriptions) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 RMTLOCNAME only allowed if DEVD(*LOC) is specified.

Purpose

The Work with Device Descriptions (WRKDEVD) command allows you to display and work with device description functions. When this command is entered, the Work with Device Descriptions display is shown. This display consists of a list of device descriptions to which you are authorized.

From the Work with Device Descriptions display, you can perform all of the available configuration functions on the device descriptions. You can create, change, copy, delete, display, and print the device descriptions.

This command also allows you to group the device descriptions that appear on the Work with Device Descriptions display. Group displays can include all of the device descriptions of only a certain type of device, or only descriptions of devices whose names start with a certain character string prefix.

Optional Parameters

DEVD

Specifies the name of the device descriptions being listed on the Work with Device Descriptions display.

***ALL:** All device descriptions are shown.

***CMN:** Communications devices are shown.

***DKT:** Diskette devices are shown.

***DSP:** Both local and remote display station devices are shown.

***LCLDSP:** Local display station devices are shown.

***RMTDSP:** Remote display station devices are shown.

***VRTDSP:** Virtual display station pass-through devices are shown.

***LOC:** Devices whose remote location name matches the name specified on the RMTLOCNAME parameter are shown. A value, other than *NONE, must be specified for the RMTLOCNAME parameter if you specify *LOC for the DEVD parameter.

***PRT:** Both local and remote printer devices are shown.

***LCLPRT:** Local printer devices are shown.

***RMTPRT:** Remote printer devices are shown.

***VRTPRT:** Virtual (passthrough) printer devices are shown.

***TAP:** Tape devices are shown.

***SNPT:** SNA pass-through devices are shown.

generic-device-name:* Specify the generic name of the device. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on

the use of generic functions, refer to "Rules for Specifying Names."

device-name: Specify name of the device descriptions to be shown.

RMTLOCNAME

Specifies the remote location name for communications devices being shown. Only device descriptions with the specified remote location name are listed on the Work with Device Descriptions display.

This parameter is required if DEVD(*LOC) is specified and is not valid if DEVD(*LOC) is not specified.

***NONE:** The descriptions shown are not for a specific location. *NONE must be specified if the DEVD parameter is other than *LOC.

generic-remote-location-name:* Specify the generic name of the remote location.

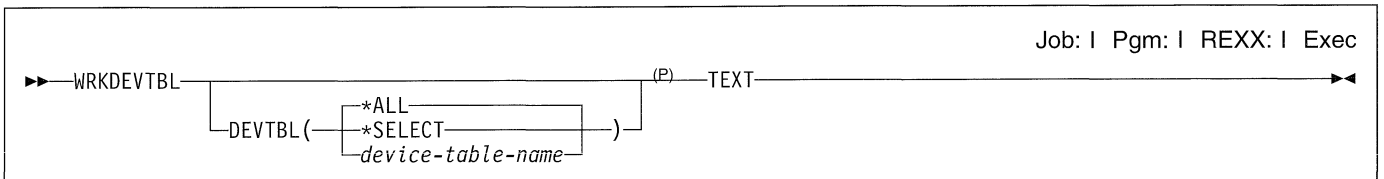
remote-location-name: Specify the remote location name of the communications devices to be shown.

Example

```
WRKDEVD  DEVD(*LCLPRT)
```

This command allows you to use the Work with Device Descriptions display of all the local printers to which you have authority.

WRKDEVTBL (Work with Device Tables) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Device Tables (WRKDEVTBL) command allows you to display and work with finance device tables and, once they are created, allows addition or deletion of device names in these tables. Several finance device tables can be defined, but each table must have a unique name.

An updated finance device table can be accessed by any finance job submitted after all changes are completed.

Restriction: Only the QFNC user profile is authorized to use this command.

Optional Parameters

DEVTBL

Specifies the name of a device table that has space for 4704 or 3624 device names.

***ALL:** The list of existing device tables is shown. From this display, you can create, change, delete, or display device tables.

***SELECT:** The list of existing device tables is shown. This value, which performs the same function as *ALL, is included for compatibility with previous releases.

device-table-name: Specify the name of the device table with which to work.

TEXT

Unused parameter provided for compatibility with previous releases.

Examples

Example 1: Working With All Finance Device Tables

```
WRKDEVTBL  DEVTBL(*SELECT)
```

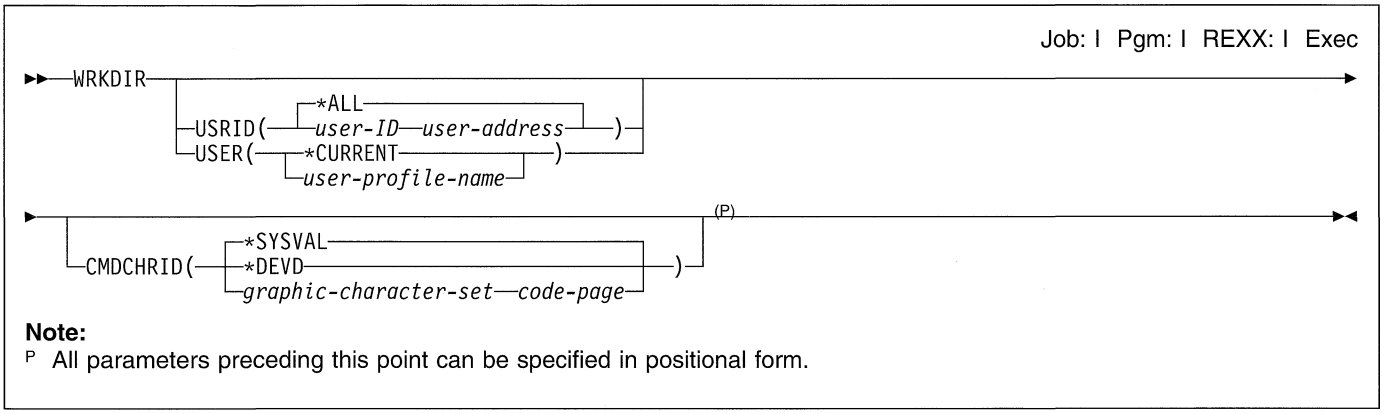
This command allows you to work with all of the finance device tables. The options are to add a new table, select an existing table for update, or create, delete, or display tables.

Example 2: Working With One Finance Device Table

```
WRKDEVTBL  DEVTBL(DEVTBL1)
```

This command allows you to work with the device table DEVTBL1. The options are to create a new table, or change, display, or delete the table.

WRKDIR (Work with Directory) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Directory (WRKDIR) command allows the user to work with a set of panels that allow viewing, adding, changing, and removing entries in the distribution directory. When the WRKDIR command is entered, the system shows either one or all of the entries in the system distribution directory, depending on the parameters specified. If the parameter specified applies to more than one directory entry, the system displays a list of directory entries. If the parameter identifies a specific directory user, the system displays a list of entries for which that user has authority.

Restriction: Users of this command must have security administrator authority (*SECADM) to update all entries in the directory. Restrictions on the data entries that can be updated apply when a nonadministrator runs this command. General access to view and print the directory is provided by the DSPDIR command.

Optional Parameters

USRID

Specifies the user ID and address of user for whom the request is made. If USRID is specified, USER cannot be specified.

***ALL:** All directory entries in the system distribution directory are shown. The entries are shown in alphabetic order by user ID and address.

Element 1: User ID

user-ID: Specify the user ID of the user for whom the directory entry is shown.

Element 2: User Address

user-ID: Specify the user address of the user for whom the directory entry is shown.

USER

Specifies, by user profile, which directory entry to display. If the user profile has no directory entries associated with it, an error message is sent. If USER is specified, USRID cannot be specified.

***CURRENT:** The user profile under which the current job is running is used.

user-profile-name: Specify the user profile of a directory entry shown. This is the 10-character profile used to sign on the system.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the *Guide to Programming Displays*.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values used to create the command parameters. Valid values range from 1 through 9999.

Element 2: Code Page

code-page: Specify the code page. Valid values range from 1 through 9999.

Examples

Example 1: User with Administrator Authority

```
WRKDIR  USRID(HURST NEWYORK)
```

Assume the user who is running this command has administrator authority. If the user ID and address of HURST NEWYORK exists in the directory, the Work with Directory

Entries panel is shown listing all entries for HURST NEWYORK.

Example 2: User with Security Administrator Authority

```
WRKDIR  USER(JONES)
```

Assume the user who is running this command has security administrator authority. If the user profile of JONES exists in the directory, the Work with Directory Entries panel displays the entry with the user profile name of JONES. Multiple entries are displayed if JONES has more than one description.

Example 3: User with Security Administrator Authority

```
WRKDIR
```

Assume the user who is running this command has security administrator authority. The Work with Directory Entries panel displays a listing of all entries in the directory.

Example 4: User Without Security Administrator Authority

```
WRKDIR
```

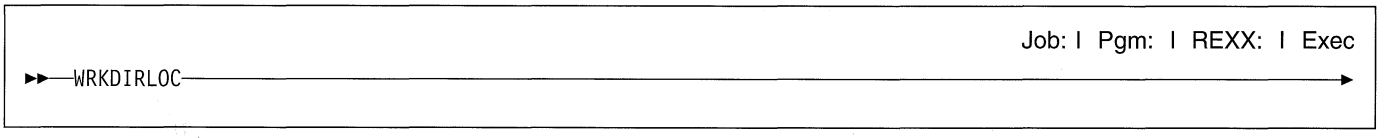
Assume the user who is running this command does not have security administrator authority. The Change Your Directory Details panel is displayed for this user. A message appears on the message line of this panel indicating that this user is authorized only to change the user's directory entry.

Additional Considerations

If the USRID parameter specifies the user ID and address of a user not currently listed in the distribution directory, an error message is returned. Likewise, if the USER parameter specifies a user profile not contained in the directory, an error message is returned.

This command allows a nonadministrator to update only certain fields for the personal directory entry. If a nonadministrator runs this command and specifies a USRID or USER parameter of someone else, an error message is displayed on the Change Your Directory Details panel indicating that the user is not authorized to update anyone else's entries.

WRKDIRLOC (Work with Directory Locations) Command



Purpose

The Work with Directory Locations (WRKDIRLOC) command provides a set of displays that allow an administrator to add, change, remove, display, print, and combine locations. When the WRKDIRLOC command is entered, the Work with Directory Locations display is shown with all the locations defined.

Restriction: The user of this command must have at least security administrator (*SECADM) authority.

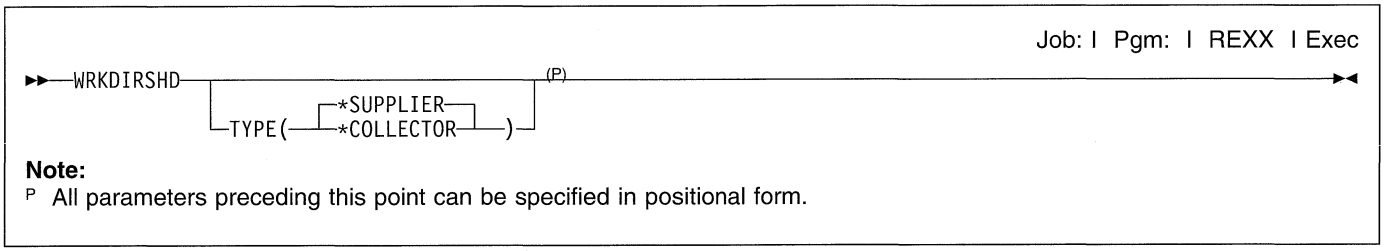
There are no parameters for this command.

Examples

```
WRKDIRLOC
```

The Work with Directory Locations display is shown. The display lists all of the locations currently defined.

WRKDIRSHD (Work with Directory Shadow Systems) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Directory Shadow Systems (WRKDIRSHD) command provides a set of displays that allows an administrator to view, add, change, and remove shadow system entries. The user can work with systems that are supplying the local system or are collecting from the local system.

Restriction: You must have security administrator (*SECADM) authority to use this command.

Optional Parameters

TYPE
Specifies the shadow systems with which the user wants to work.

***SUPPLIER:** The user is allowed to work with the systems supplying directory data to the local system. The Work with Directory Shadow Suppliers display is shown.

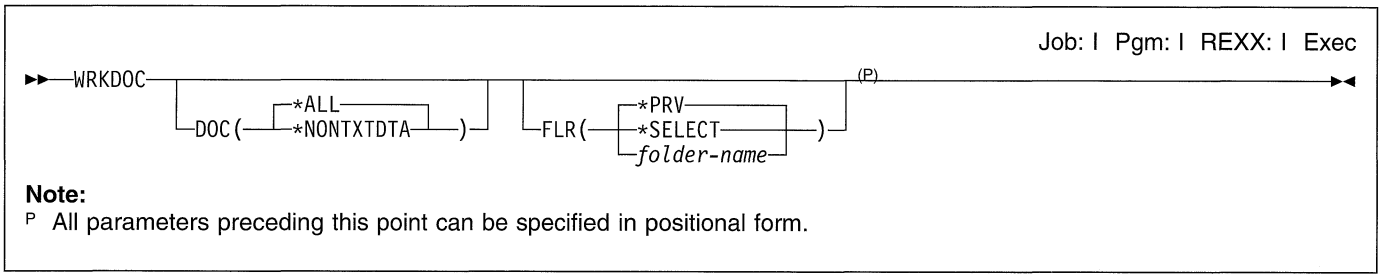
***COLLECTOR:** The user is allowed to work with the systems collecting from the local system. The Work with Directory Shadow Collectors display is shown.

Example

```
WRKDIRSHD  TYPE(*SUPPLIER)
```

The Work with Directory Shadow Suppliers display is shown, listing all of the shadow suppliers currently on the system.

WRKDOC (Work with Documents) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Documents (WRKDOC) command allows you to display and work with the word processing function of Office/Vision/400 to show the Work with Documents display or the Work with Nontext Document Data display.

From the Work with Documents display, you can select an option to: create, revise, copy, delete, view, print, rename, describe, print with options, send, spell check, file, paginate, and specify security levels on documents.

From the Work with Nontext Document Data display, you can select an option to: copy, delete, and rename nontext data, such as charts and images.

More information on working with documents is in the *Using OfficeVision/400* Word Processing*.

Optional Parameters

DOC

Specifies which display is shown.

***ALL:** The Work with Documents display is shown.

***NONTXTDTA:** The Work with Nontext Document Data display is shown.

FLR

Specifies the name of the folder used on the Work with Documents display or Work with Nontext Document Data display.

***PRV:** The name used in the previous session is used.

***SELECT:** Displays a list of folders from which you can select a folder.

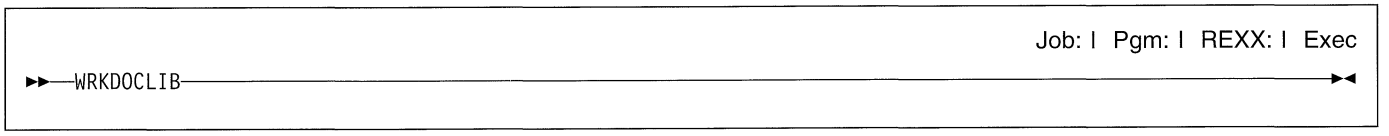
folder-name: Specify the name of the folder to work with on the specified display.

Example

```
WRKDOC  DOC(*ALL)  FLR(*SELECT)
```

This command allows you to utilize the Work with Documents display and to show a list of folders from which to select the working folder.

WRKDOCLIB (Work with Document Libraries) Command



Purpose

The Work with Document Libraries (WRKDOCLIB) command allows you to display and work with the Document Interchange Architecture (DIA) library services available on remote systems in the network. Each remote document library for which services are obtained can be configured. This includes indicating the level of DIA services provided by the remote system and associating the remote document library with a SNADS queue on which work requests are placed.

Restriction: You must have security officer or security administrator authority to use this command.

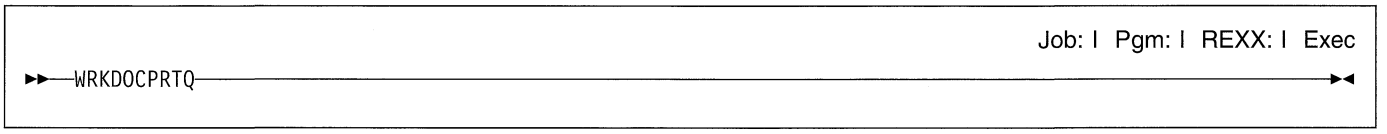
There are no parameters for this command.

Example

```
WRKDOCLIB
```

This command allows the user to work with the Work with Document Libraries display.

WRKDOCPRTQ (Work with Document Print Queue) Command



Purpose

The Work with Document Print Queue (WRKDOCPRTQ) command allows you to display and work with the word processing function of OfficeVision/400 to show the Work with Documents to Be Printed display. Users can manage printed output from this display.

More information on word processing is in the *Using OfficeVision/400* Word Processing*.

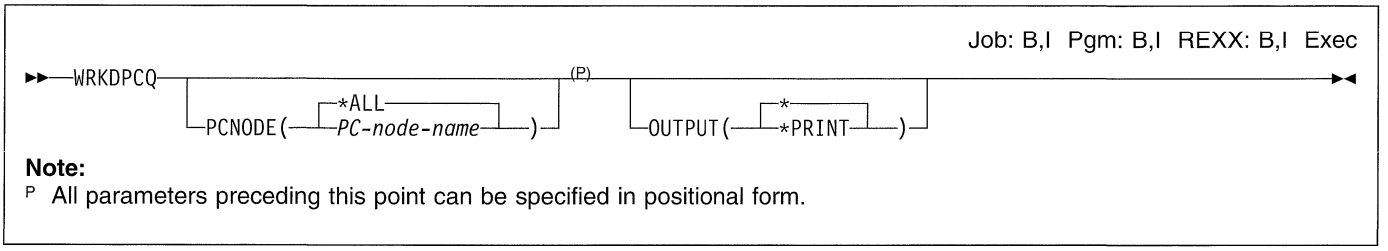
There are no parameters for this command.

Example

```
WRKDOCPRTQ
```

This command allows you to work with the Work with Documents to be Printed display.

WRKDPCQ (Work with DSNX/PC Distribution Queues) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Distributed System Node Executive/Personal Computer Distribution Queues (WRKDPCQ) command allows the user to display and work with some simple operations on the DSNX/PC distribution queues. These queues are where Distributed System Node Executive (DSNX) distributions that are bound for a PC (locally attached to the AS/400 system and configured in the system directory as a DSNX-PC node) are held. A PC running DSNX-PC initiates the DS-SEND function that sends the queued distributions to that requesting PC.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR and QSYSOPR user profiles have private authorities to use the command.

Optional Parameters

PCNODE

Specifies the names of PC nodes for which queue entries are shown.

*ALL: All the PC nodes that currently have queued entries are shown.

PC-node-name: Specify the name of the PC node to show.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

_*:_ Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

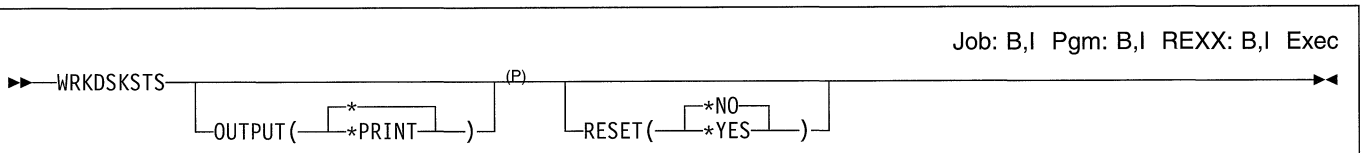
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKDPCQ
```

This command allows the user to work with the Work with DSNX/PC Distribution Queues display.

WRKDSKSTS (Work with Disk Status) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Disk Status (WRKDSKSTS) command allows the user to display and work with performance and status information for the disk units on the system.

Optional Parameters

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

RESET

Specifies whether the disk statistics are reset.

***NO:** The disk statistics are not reset. The measurement time interval is extended if a previous display disk status command is active in the current job.

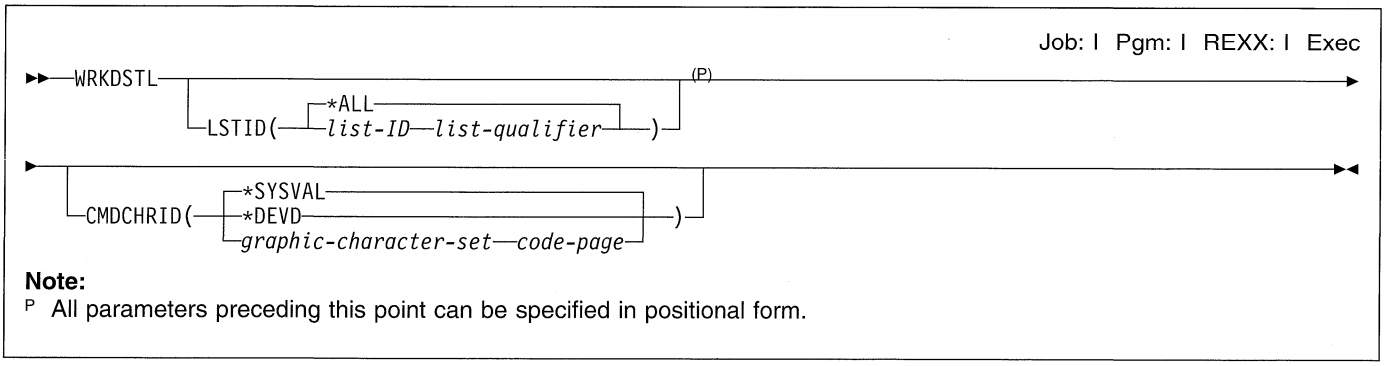
***YES:** The disk statistics are reset. A measurement time interval of zero (0) is used.

Example

```
WRKDSKSTS  OUTPUT(*PRINT)
```

This command allows the user to work with the disk information and to print the job's spooled output.

WRKDSTL (Work with Distribution Lists) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Distribution Lists (WRKDSTL) command allows you to view, create, add entries to, remove entries from, and delete distribution lists. A distribution list is a list of directory entries used to simplify the sending of distributions to a group of users. When the WRKDSTL command is entered, the system shows one, some, or all of the distribution lists in the system directory, depending on the value specified on the LSTID parameter. If *ALL is specified on the LSTID parameter, all distribution lists are shown. If a specific list ID is specified on the LSTID parameter, a specific distribution list is shown.

Restriction: Users must have security administrator authority (*SECADM) to change or delete another user's distribution list. Users can change and delete distribution lists that they create without any special authorities; any user can create a distribution list. All users can view all distribution lists.

Optional Parameters

LSTID

Specifies which distribution lists to be shown. If a list ID not currently contained in the distribution directory is specified on the LSTID parameter, an error message is returned.

***ALL:** All distribution lists in the system distribution directory are shown in alphabetical order by list ID.

Element 1: List Identifier

list-identifier: Specify the list identifier (ID) of a distribution list.

Element 2: List Qualifier

list-qualifier: Specify the list qualifier of the distribution list.

Note: The distribution list identifier/list qualifier is in two parts, separated by at least one space. If any lowercase characters are specified, the system changes them to uppercase.

The naming rules for the two-part list ID are identical to the rules for the user ID and address. A complete description of these rules is in the *Distribution Services Network Guide*.

CMDCHRID

Specifies the character identifier (graphic character set and code page) for data being specified as parameter values on this command. This character identifier (CHRID) is related to the display device used to specify the command. More information about CHRID processing is in the *Guide to Programming Displays*.

***SYSVAL:** The system determines the graphic character set and code page values for the command parameters from the QCHRID system values.

***DEVD:** The system determines the graphic character set and code page values for the command parameter from the display device description where the command is entered. This option is valid only when specified from an interactive job. If this value is specified in an interactive CL program or a batch job, an error message is sent.

Element 1: Character Set

graphic-character-set: Specify the graphic character set values used to create the command parameters. Valid values range from 1 through 9999.

Element 2: Code Page

code-page: Specify the code page. Valid values range from 1 through 9999.

Examples

Example 1: Showing a Distribution List

```
WRKDSTL  LSTID(WILL DISTLIST)
```

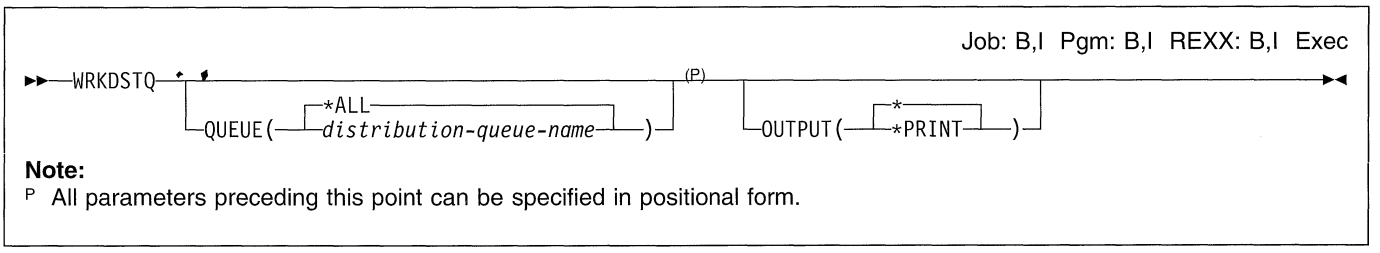
This command shows the Work with Distribution Lists display with one entry, WILL DISTLIST. This example assumes that WILL DISTLIST exists.

Example 2: Showing All Distribution Lists

```
WRKDSTL
```

This command shows the Work with Distribution Lists display with a list of all distribution lists in the distribution directory.

WRKDSTQ (Work with Distribution Queue) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Distribution Queue (WRKDSTQ) command allows the user to display and work with the distribution requests on the systems network architecture distribution services (SNADS) distribution queues. A detailed description of SNADS is in the *Communications: Distribution Services Network Guide*.

Distribution queue names are translated to the graphic character set and code page 930 500, using the job's coded character set identifier (CCSID).

Restrictions:

1. This command is shipped with public *EXCLUDE authority, and the QPGMR and QSYSOPR user profiles have private authorities to use the command.
2. Before this command is run for the first time, the QSNADS subsystem must have been previously started to create the internal SNADS objects that this command uses.
3. Messages that report errors about distribution queues may display or print different characters than the user entered for the distribution queue name because of internal system transformations. Similarly (depending on the language used for the work station), the internal value for a distribution queue name may differ from the characters shown on the Work with Distribution Queue (WRKDSTQ) command. An error may be reported if the character-string value specified for the QUEUE parameter does not match the rules for an internal distribution queue value or if it does not match the internal value for any defined distribution queue (ignoring case differences).

Optional Parameters

QUEUE
Specifies the name of the distribution queue being shown or printed. The queue must have been previously configured using the Configure Distribution Services (CFGDSTSRV) command or the Add Distribution Queue (ADDDSTQ) command.

***ALL:** The normal and high priority of all distribution queues are shown or printed in alphabetical order.

distribution-queue-name: Specify the name of the distribution queue to show or print. Both normal and high priority portions of the specified distribution queue are shown or printed.

OUTPUT
Specifies whether the output from the command is displayed at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

Examples

Example 1: Working With All Distribution Queues

```
WRKDSTQ
```

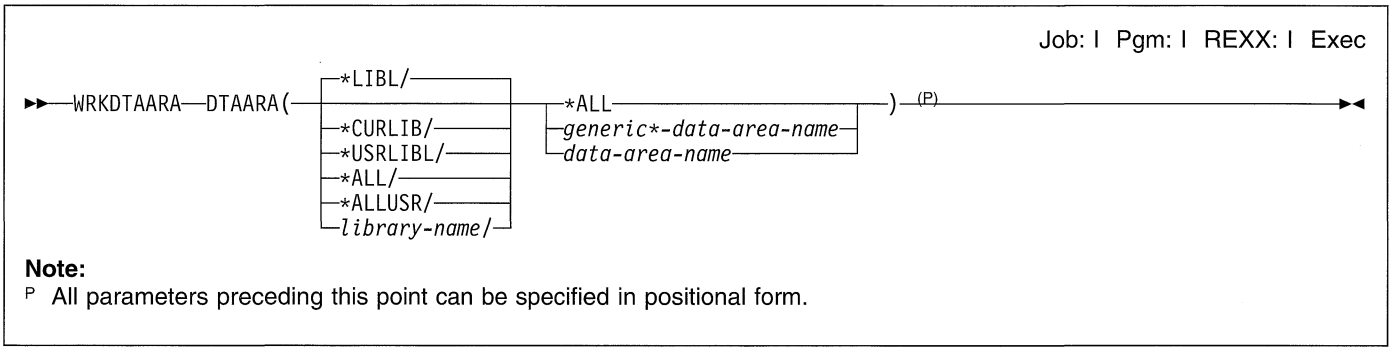
This command allows the user to work with the status and contents of all distribution queues. The normal and high priority portions of each distribution queue are shown or printed.

Example 2: Printing Information

```
WRKDSTQ  OUTPUT(*PRINT)
```

This command prints information on all distribution queues. The status of the normal and priority portions of the distribution queues are printed followed by a list of the distribution requests on the normal and high priority portions of each distribution queue.

WRKDTAARA (Work with Data Areas) Command



Note:
| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Data Areas (WRKDTAARA) command allows the user to display and work with a list of data areas from one or more libraries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the data areas to which you have some authority will be shown on the display.
3. To perform operations on the data areas, you must have USE authority to the command used by the operation, and the appropriate authority to the data areas on which the operation is to be performed.

Required Parameters

DTAARA

Specifies a list of data areas in the libraries that are shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the data areas.

| The name of the data areas can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All data areas in the libraries identified in the library qualifier are shown.

generic-data-area-name:* Specify the generic name of
| the data area. A generic name is a character string of
| one or more characters followed by an asterisk (*); for
| example, ABC*. The asterisk substitutes for any valid
| characters. A generic name specifies all objects with
| names that begin with the generic prefix for which the
| user has authority. If an asterisk is not included with the
| generic (prefix) name, the system assumes it to be the
| complete object name. If the complete object name is
| specified, and multiple libraries are searched, multiple
| objects can be worked with only if *ALL or *ALLUSR
| library values can be specified for the name. For more
| information on the use of generic functions, refer to
| "Rules for Specifying Names."

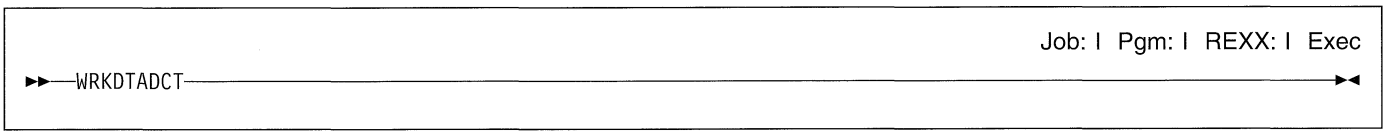
data-area-name: Specify the name of the data area that is shown.

Example

```
WRKDTAARA   DTAARA(LIB01/ABC*)
```

This command allows you to display and work with a list of data areas beginning with ABC stored in library LIB01.

WRKDTADCT (Work with Data Dictionaries) Command



Purpose

The Work with Data Dictionaries (WRKDTADCT) command allows you to work with the Interactive Data Definition Utility (IDDU) Work with Data Dictionaries display on which you can select an option to create, change, delete, or print the contents of a data dictionary.

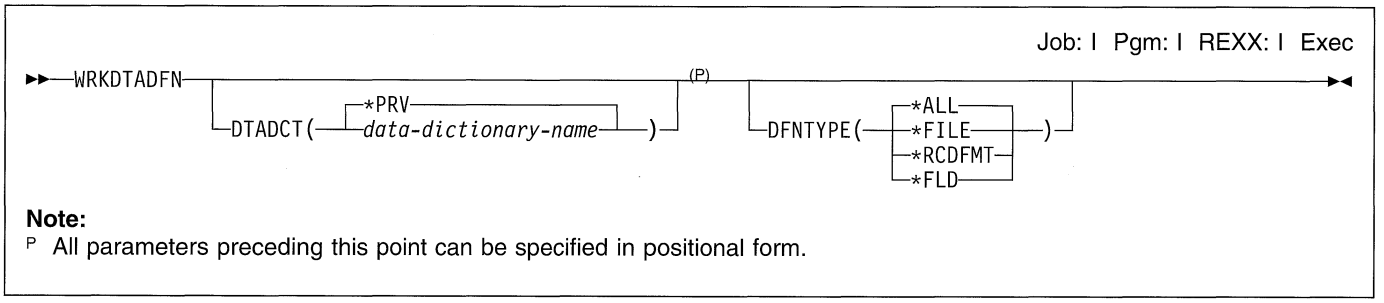
There are no parameters for this command.

Example

```
WRKDTADCT
```

This command allows you to work with the Work with Data Dictionaries display.

WRKDTADFN (Work with Data Definitions) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Data Definitions (WRKDTADFN) command allows you to create, change, copy, delete, print, or rename definitions, or display where definitions are used. If the definition type is not specified, the Interactive Data Definition Utility (IDDU) Select Definition Type display is shown. The dictionary and type of definition to process is selected from this display.

Optional Parameters

DTADCT

Specifies the data dictionary being used.

***PRV:** The name used in the previous session is used.

data-dictionary-name: Specify the data dictionary name to use.

DFNTYPE

Specifies the data definition type being used.

***ALL:** Users select the definition type and data dictionary from a list of all the data dictionaries and definition types.

***FILE:** Users work with the definition type from a list of all file definitions in the data dictionary specified.

***RCDFMT:** Users work with record format definitions in the data dictionary specified.

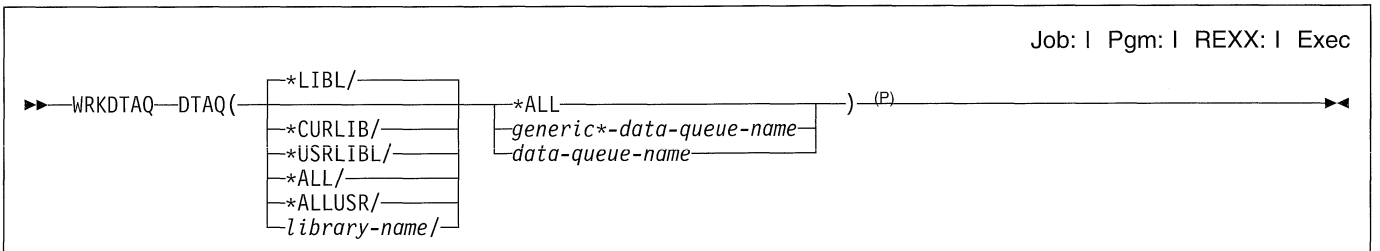
***FLD:** Users work with field definitions in the data dictionary specified.

Example

```
WRKDTADFN   DFNTYPE(*FILE)
```

This command allows you to work with file definitions in the data dictionary you worked with last.

WRKDTAQ (Work with Data Queues) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Data Queues (WRKDTAQ) command allows the user to display and work with a list of data queues from one or more libraries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the data queues to which you have some authority will be shown on the display.
3. To perform operations on the data queues, you must have USE authority to the command used by the operation, and the appropriate authority to the data queues on which the operation is to be performed.

Required Parameters

DTAQ

Specifies a list of data queues in the libraries that are shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the data queues.

The name of the data queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries,* and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All data queues in the libraries identified in the library qualifier are shown.

generic-data-queue-name:* Specify the generic name of the data queue. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

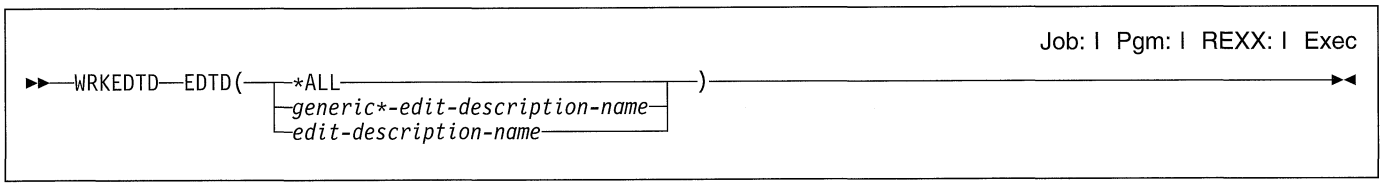
data-queue-name: Specify the name of the data queue that is shown.

Example

```
WRKDTAQ  DTAQ(LIB01/ABC*)
```

This command allows you to display and work with a list of data queues beginning with ABC stored in library LIB01.

WRKEDTD (Work with Edit Descriptions) Command



Purpose

The Work with Edit Descriptions (WRKEDTD) command allows the user to display and work with a list of edit descriptions.

Restrictions:

1. Only the edit descriptions to which you have some authority will be shown on the display.
2. To perform operations on the edit descriptions, you must have USE authority to the command used by the operation, and the appropriate authority to the edit descriptions on which the operation is to be performed.

Required Parameters

EDTD

Specifies a list of edit descriptions in the libraries that are shown.

***ALL:** All edit descriptions in the libraries identified in the library qualifier are shown (except those edit descriptions for which the user does not have some authority).

generic-edit-description-name:* Specify the generic name of the edit description. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

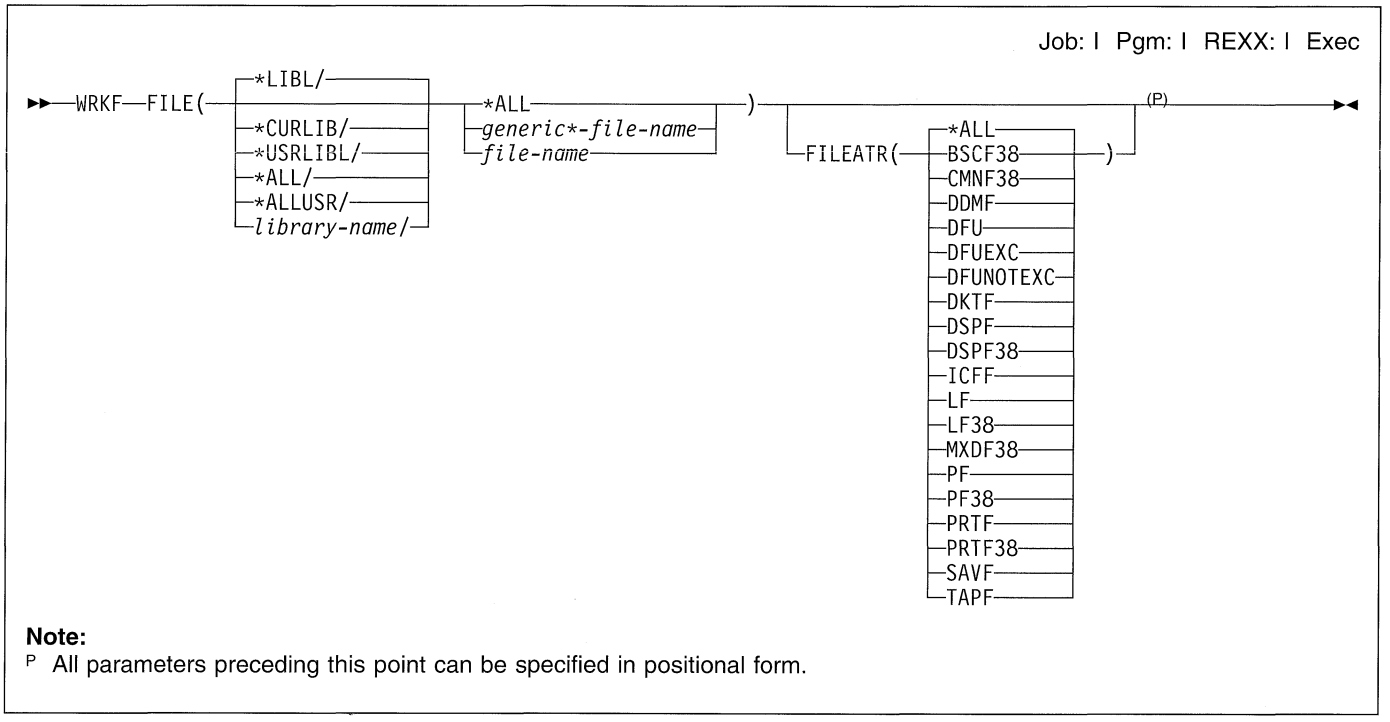
edit-description-name: Specify the name of the edit description to be shown.

Example

```
WRKEDTD  EDTD(ABC*)
```

This command allows you to display and work with a list of edit descriptions beginning with ABC.

WRKF (Work with Files) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Files (WRKF) command allows you to copy, delete, display descriptions, save, and restore files that you have authority to use.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the files to which you have some authority will be shown on the display.
3. To perform operations on the files, you must have USE authority to the command used by the operation, and the appropriate authority to the files on which the operation is to be performed.

Required Parameters

FILE

Specifies which files in the libraries are shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the files.

| The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All files in the libraries identified in the library qualifier are shown.

generic-file-name:* Specify the generic name of the file
| name. A generic name is a character string of one or
| more characters followed by an asterisk (*); for example,
| ABC*. The asterisk substitutes for any valid characters.
| A generic name specifies all objects with names that

begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

file-name: Specify the name of the file that is shown. If the library qualifier is *ALL, *ALLUSR, or a library name, all files of the specified types to which the user has some authority (for example, operational authority), and that are in the specified libraries, are shown.

Optional Parameters

FILEATR

Specifies the type of file of which the attributes are shown.

***ALL:** The attributes of all files are shown.

BSCF38: The attributes of the binary synchronous communication file for a System/38 are shown.

CMNF38: A file that is used to read data from, or write data to a logical unit (LU1) or advanced program-to-program communications (APPC) device and that allows the user to define the format of the data on the logical unit or device. A communications (CMNF38) file is a device file that is either created in the System/38 environment or migrated from a System/38 to support a communications device.

DDMF: The attributes of the distributed data management (DDM) file are shown.

DFU: The attributes of a display file created by the AS/400 system data file utility are shown.

DFUEXC: The attributes of a display file created by the System/38 data file utility (DFU), which can be run by using the System/38 data file utility, is shown.

DFUNOTEXC: A display file that is created by the System/38 data file utility (DFU), but cannot be run by using the System/38 data file utility is shown.

DKTF: The attributes of diskette files are shown.

DSPF: The attributes of display files are shown.

DSPF38: The attributes of display files for a System/38 are shown.

ICFF: The attributes of communications files are shown.

LF: The attributes of logical files are shown.

LF38: The attributes of logical files for a System/38 are shown.

MXDF38: The attributes of mixed files for a System/38 are shown.

PF: The attributes of physical files are shown.

PF38: The attributes of physical files for a System/38 are shown.

PRTF: The attributes of printer files are shown.

PRTF38: The attributes of printer files for a System/38 are shown.

SAVF: The attributes of save files are shown.

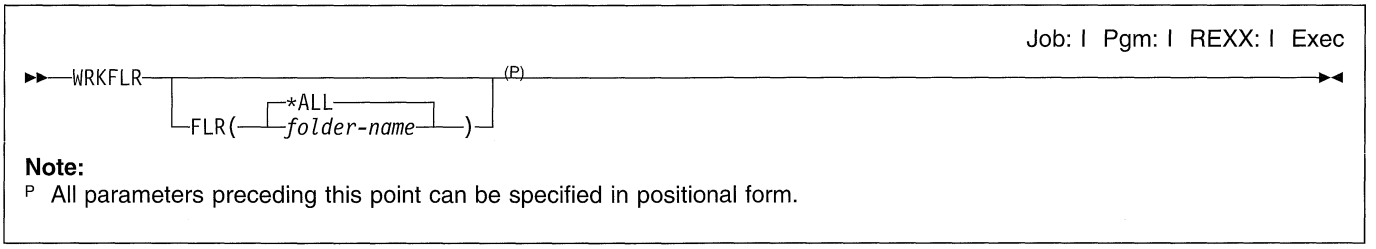
TAPF: The attributes of tape files are shown.

Example

```
WRKF  FILE(X/PAY)
```

This command allows you to work with the file named PAY, for which the user has some authority, and is located in library X.

WRKFLR (Work with Folders) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Folders (WRKFLR) command allows you to display and work with the word processing function of OfficeVision/400 to show the Work with Folders display. From this display, you can optionally create, delete, rename, describe entries; put security on a folder; work with documents; or work with folder authority.

More information on working with folders is in *Using OfficeVision/400* Word Processing*.

Optional Parameters

FLR

Specifies the name of the folder used on the Work with Folders display. The list shown is always the list of folders to which you have authority.

***ALL:** The list of all of the root level (first level) folders is displayed.

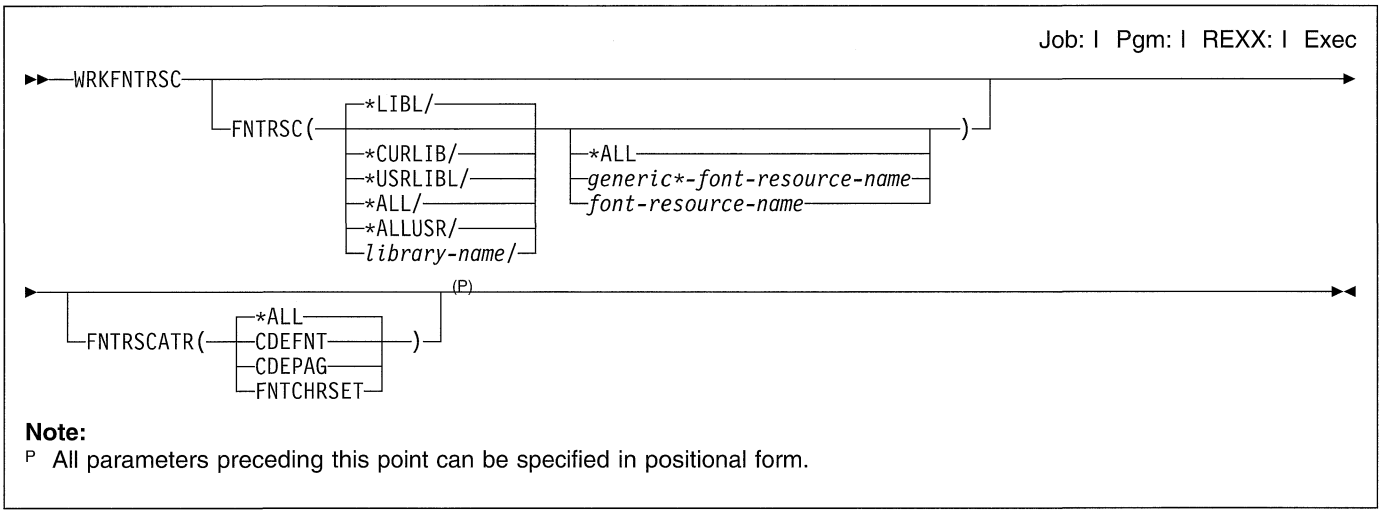
folder-name: Specify the name of the root level folder that contains the folders to display.

Example

```
WRKFLR  FLR(*ALL)
```

This command allows you to utilize the Work with Folders display. A list of all folders the user is authorized to use is shown.

WRKFNTRSC (Work with Font Resources) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Font Resources (WRKFNTRSC) command allows the user to display and work with all of the font resources from the system library, the user libraries, or both.

Optional Parameters

FNTRSC

Specifies the qualified name of the font resource.

| The name of the font resource can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All font resources in the libraries identified in the library qualifier are shown. Only font resources for which the user has some authority can be shown.

generic-font-resource-name:* Specify the generic name of the font resource. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

font-resource-name: Specify the name of the font resources that are shown.

FNTRSCATR

Specifies the type of font resource whose attributes are shown.

***ALL:** All types of font resources and all attributes are shown.

CDEFNT: A list of coded fonts is shown.

CDEPAG: A list of coded pages is shown.

FNTCHRSET: A list of font character sets is shown.

Examples

Example 1: Searching for Font Resources

```
WRKFNTRSC   FNTRSC(*ALL/GOTHIC*)
  FNTRSCATR(*ALL)
```

This command searches all libraries for the font resources whose first characters are GOTHIC. All font resource types are shown.

Example 2: Searching for Font Resources

```
WRKFNTRSC   FNTRSC(MYLIB/GOTHIC*)
  FNTRSCATR(FNTCHRSET)
```

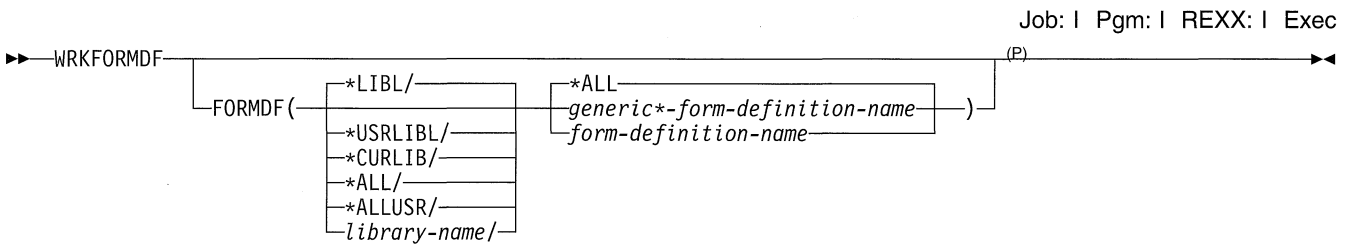
This command searches library MYLIB for all font resources beginning with GOTHIC. Only font resources with attribute FNTCHRSET are shown.

Example 3: Searching for Font Resources

```
WRKFNTRSC   FNTRSC(MYLIB/CODEPG3)   FNTRSCATR(CDEPAG)
```

This command searches the library MYLIB for a font resource with the name CODEPG3 and the attribute CDEPAG.

WRKFORMDF (Work with Form Definitions) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Form Definitions (WRKFORMDF) command allows you to work with all of the form definitions from the system or user libraries (or both).

Optional Parameters

FORMDF

Specifies the qualified name of the form definition.

| The name of the form definition can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All form definitions in the libraries identified in the library qualifier are listed. Only those form definitions for which the user has some authority are shown.

generic-form-definition-name:* Specify the generic name | of the form definition. A generic name is a character | string of one or more characters followed by an asterisk | (*); for example, ABC*. The asterisk substitutes for any | valid characters. A generic name specifies all objects | with names that begin with the generic prefix for which | the user has authority. If an asterisk is not included with | the generic (prefix) name, the system assumes it to be | the complete object name. If the complete object name | is specified, and multiple libraries are searched, multiple | objects can be worked with only if *ALL or *ALLUSR | library values can be specified for the name. For more | information on the use of generic functions, refer to | "Rules for Specifying Names."

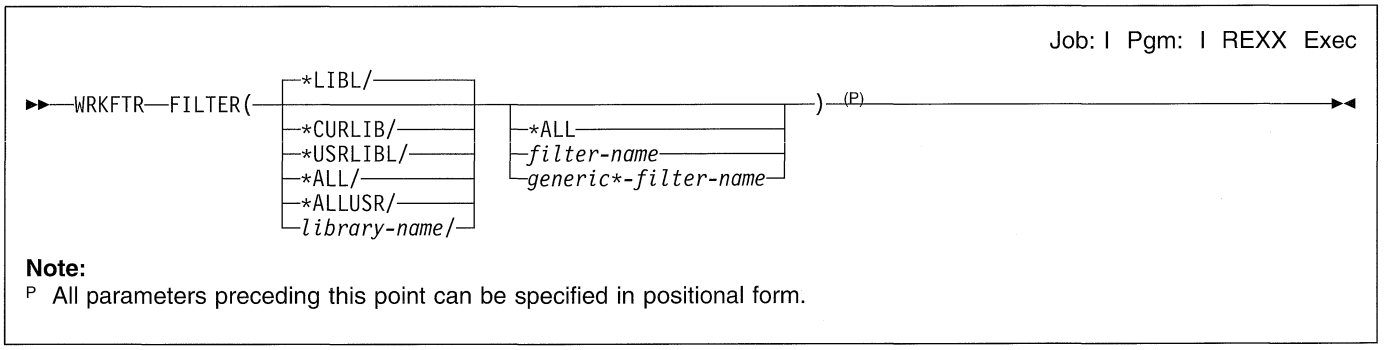
form-definition-name: Specify the name of the form definition that is listed.

Example

```
WRKFORMDF  FORMDF(*CURLIB/FORMDF1)
```

This command searches the current library for the form definition FORMDF1. If FORMDF1 does not exist, the WRKFORMDF panel shows a message indicating that an object matching the specified name cannot be found.

WRKFTR (Work with Filters) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Filters (WRKFTR) command allows you to work with and print a list of filters, to change and delete specified filters, to work with selection and action entries contained in specified filters, and to create new filters.

Restrictions:

- Only the libraries to which you have *READ authority are searched.
- Only the filters to which you have authority are shown.
- To perform operations on the filters, you must have *USE authority to the command used by the operation, and the appropriate authority to the filters on which the operation is to be performed.

Required Parameters

FILTER

Specifies the qualified name of the filter that is shown.

The name of the filter can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All filters in the specified library are listed.

filter-name: Specify the name of the filter that is shown.

generic-filter-name:* Specify the generic name of the filter. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

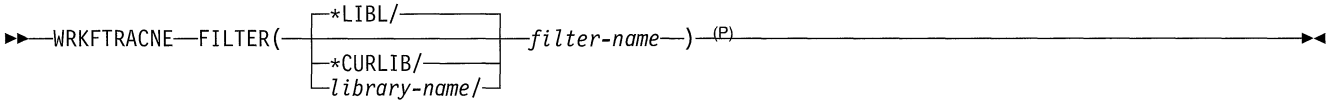
Example

```
WRKFTR  FILTER(MYLIB/MY*)
```

This command shows a list of filters whose names begin with MY in library MYLIB. From the list shown, you can change, delete, or work with the entries in any or all of the filters shown. You can also create a new filter.

WRKFTRACNE (Work with Filter Action Entries) Command

Job: I Pgm: I REXX Exec



Note:

(P) All parameters preceding this point can be specified in positional form.

Purpose

The Work with Filter Action Entries (WRKFTRACNE) command allows you to display, add, change, copy, print, rename, or remove action entries in a filter.

Required Parameters

FILTER

Specifies the qualified name of the filter which contains the action entries.

The name of the filter can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

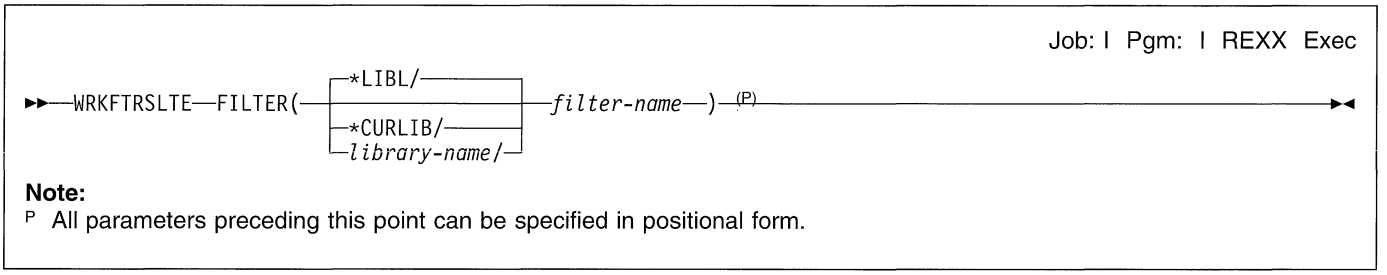
filter-name: Specify the name of the filter.

Example

```
WRKFTRACNE  FILTER(MYLIB/MYFILTER)
```

This command allows you to work with the action entries in filter MYFILTER in library MYLIB.

WRKFTRSLTE (Work with Filter Selection Entries) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Filter Selection Entries (WRKFTRSLTE) command allows you to display, add, change, copy, print, remove, or move selection entries in a filter.

Required Parameters

FILTER

Specifies the qualified name of the filter which contains the selection entries.

The name of the filter can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

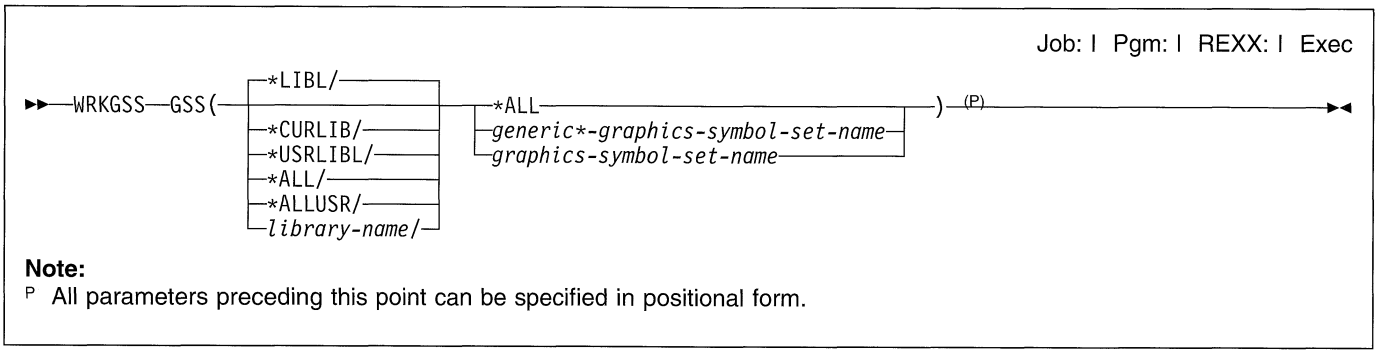
filter-name: Specify the name of the filter.

Example

```
WRKFTRSLTE  FILTER(MYLIB/MYFILTER)
```

This command allows you to work with the selection entries in filter MYFILTER in library MYLIB.

WRKGSS (Work with Graphics Symbol Sets) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Graphics Symbol Sets (WRKGSS) command allows the user to display and work with a list of graphics symbol sets from one or more libraries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the graphics symbol sets to which you have some authority will be shown on the display.
3. To perform operations on the graphics symbol sets, you must have USE authority to the command used by the operation, and the appropriate authority to the graphics symbol sets on which the operation is to be performed.

Required Parameters

GSS

Specifies the qualified name of the graphics symbol sets that are listed. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the graphics symbol sets.

| The name of the graphics symbol set can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All graphics symbol sets in the libraries identified in the library qualifier are shown.

generic-graphics-symbol-set-name:* Specify the generic | name of the graphics symbol set. A generic name is a | character string of one or more characters followed by | an asterisk (*); for example, ABC*. The asterisk substi- | tutes for any valid characters. A generic name specifies | all objects with names that begin with the generic prefix | for which the user has authority. If an asterisk is not | included with the generic (prefix) name, the system | assumes it to be the complete object name. If the com- | plete object name is specified, and multiple libraries are | searched, multiple objects can be worked with only if | *ALL or *ALLUSR library values can be specified for the | name. For more information on the use of generic func- | tions, refer to "Rules for Specifying Names."

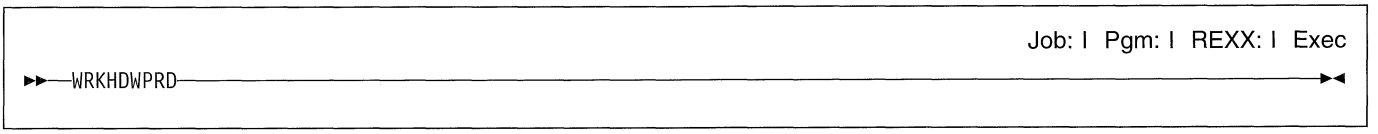
graphics-symbol-set-name: Specify the name of the graphics symbol set that is shown.

Example

```
WRKGSS  GSS(LIB01/ABC*)
```

This command allows you to display and work with a list of graphics symbol sets beginning with ABC stored in library LIB01.

WRKHDWPRD (Work with Hardware Products) Command



Purpose

The Work with Hardware Products (WRKHDWPRD) command allows you to show a menu on which you can work with, copy, or replace the system configuration information.

The Work with Hardware Products menu option 1, Work with system configuration, is chosen whenever new I/O feature cards, disk units, diskette units, or tape units are being added to, or existing objects are being moved within, the system. If any of these objects have been added or moved, you are prompted for location information. If a new frame or card enclosure is found to exist, you are prompted for its attributes.

If there have been no additions or movements of disks, diskettes, tape units, or I/O feature cards, the Work with System Configuration display is shown.

From this display, you can do the following:

- Change items
- Delete items
- Print the system configuration list
- Add filler panels
- Display location information, serial numbers and resource names, and part numbers and addresses
- Work with cables
- Reserve frame space
- Change system type and number

The Work with Hardware Products menu option 2, Copy system configuration, is chosen to copy the current system configuration list to a user-specified file.

The Work with Hardware Products menu option 3, Replace system configuration, is chosen to replace the current system configuration list with information from an existing user-specified file.

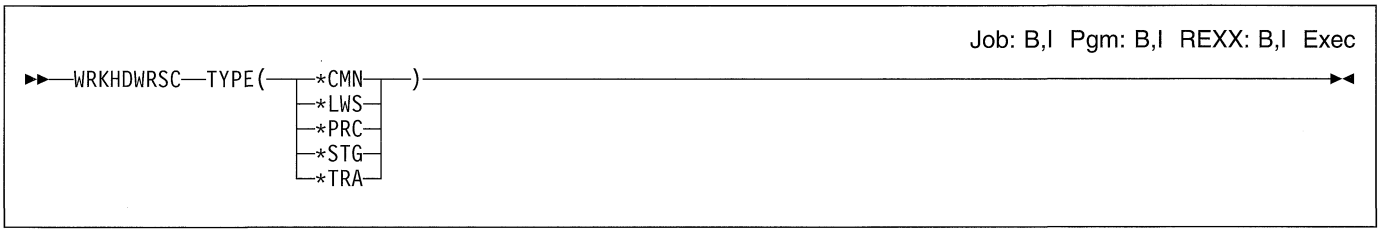
There are no parameters for this command.

Example

```
WRKHDWPRD
```

This command allows you to utilize the Work with Hardware Products display.

WRKHDWRSC (Work with Hardware Resources) Command



Purpose

The Work with Hardware Resources (WRKHDWRSC) command allows the user to manage the hardware on the system. It allows the user to work with:

- Storage
- Processors
- Communications
- Local work stations, and
- Token-ring local area network (LAN) adapters

For storage, processors, communications, and local work stations, the user can review status and related configuration descriptions, determine which resources can be configured, and determine which devices have configuration descriptions already created.

For token ring LAN adapters, the user can create a resource entry in the system or update the information. Token ring LAN adapter information consists of adapter name, adapter address, and adapter description. The Work with Token Ring Adapter (WRKTRA) command allows the user to update token ring adapter information for adapters reporting on a specified line.

Required Parameters

TYPE

Specifies the type of hardware resource with which to work.

***CMN:** The Work with Communication Resources display is shown.

***LWS:** The Work with Local Work Station Resources display is shown.

***PRC:** The Work with Processor Resources display is shown.

***STG:** The Work with Storage Resources display is shown.

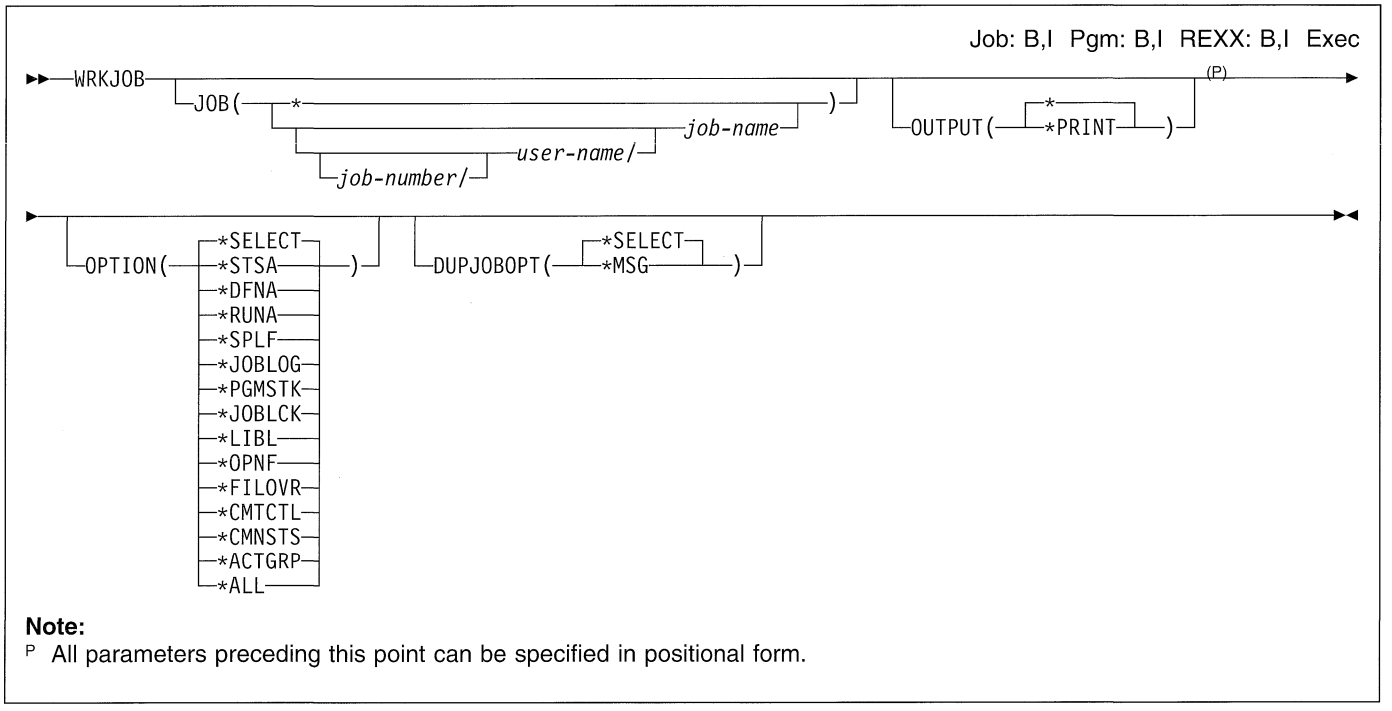
***TRA:** The Work with TRLAN Adapter Information display is shown.

Example

```
WRKHDWRSC  TYPE(*CMN)
```

This command allows the user to add, change, remove, or display information about local communication resources on the system and associated configuration objects. The Work with Communication Resources display shows all communication input/output processors (IOPs), input/output adapters (IOAs), and ports installed on the system.

WRKJOB (Work with Job) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Job (WRKJOB) command allows the user to work with or change information concerning a job.

A user can work with or change the following:

- Job status attributes
- Job definition attributes
- Job run attributes
- Spooled file information
- Job log information
- Call stack information
- Job lock information
- Library list information
- Open file information
- File override information
- Commitment control status
- Communications status
- Activation group information

The information for the following options can be shown only when the job is active: job run attributes, call stack information, job lock information, library list information, job log information, open file information, file override information, commitment control status, and communication status. Job status attributes, job definition attributes, and spooled file information can be shown whether the user's job is on the job queue, on an output queue, or active in the system. However, the job is not considered to be in the system until all of its input has been completely read in; only then is an entry placed on the job queue.

Restrictions:

1. Unless a user has special job control (*JOBCTL) authority, or unless another job has the same user name specified, only the user's job can be shown.
2. Activation group information for a job cannot be shown if the job is being held when this command is run.

Optional Parameters

JOB

Specifies the name of the job whose information the user is working with. If no job qualifier is given, all of the jobs currently in the system are searched for the simple job name. If duplicates of the specified name are found, a list of messages containing the qualified job names of all duplicates is shown.

A job identifier is a special value or a qualified name with up to three elements. For example:

```
*
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

*: The job whose information is shown is the job where this work with command is issued.

job-name: Specify the name of the job whose information is to be displayed.

user-name: Specify the name of the user of the job whose information is to be displayed.

job-number: Specify the number of the job whose information is to be displayed.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

OPTION

Specifies the job information with which the user is working. Only one value may be selected.

***SELECT:** A list of options is shown that allows a user to select the information with which to work.

***STSA:** The status attributes associated with the job are shown.

***DFNA:** The definition attributes associated with the job are shown.

***RUNA:** The run attributes associated with the job are shown.

***SPLF:** The spooled files associated with the job are shown.

***JOBLOG:** The job log associated with the job is shown. The job log is printed or shown depending on what is specified for the OUTPUT parameter. This value is not shown when either OPTION(*SELECT) or OPTION(*ALL) is specified.

***PGMSTK:** The call stack associated with the job is shown.

***JOBLCK:** The job locks associated with the job are shown.

***LIBL:** The library list associated with the job is shown.

***OPNF:** The open files associated with the job are shown.

***FILOVR:** The file overrides associated with the job are shown.

***CMTCTL:** The commit control status of the job is shown.

***CMNSTS:** The communications status of the job is displayed.

***ACTGRP:** The activation groups associated with the job are shown.

***ALL:** All job information associated with the job is shown.

DUPJOBOPT

Specifies the action taken when duplicate jobs are found by this command.

***SELECT:** The selection display is shown when duplicate jobs are found during an interactive session. Otherwise, a message is issued.

***MSG:** A message is issued when duplicate jobs are found.

Examples

Example 1: Printing the Job's Information

```
WRKJOB  JOB(SMITH/PAYROLL)  OUTPUT(*PRINT)
```

This command allows the user to print information for the job named PAYROLL submitted by the user named SMITH to the job's output spooling queue for printing.

Example 2: Working with the Current Job's Spooled Output

```
WRKJOB  OPTION(*SPLF)
```

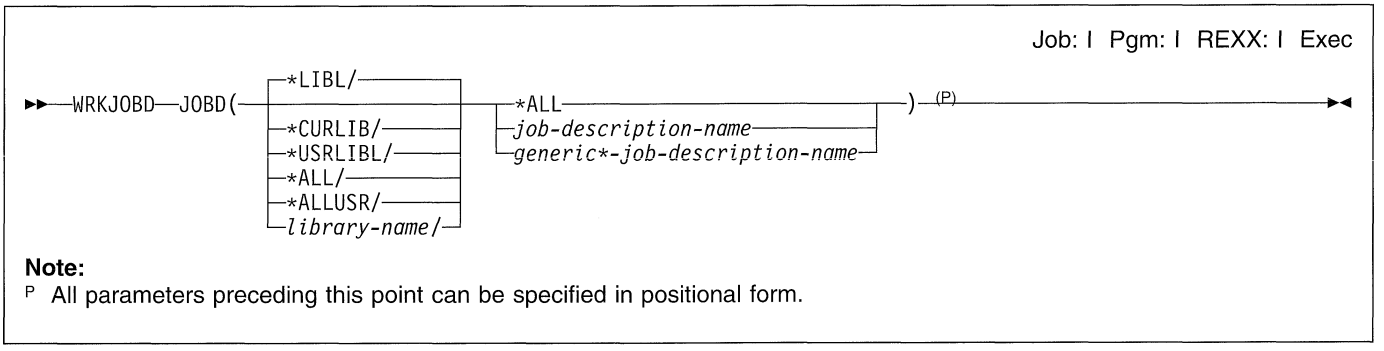
This command allows the user to work with the spooled output for the current job.

Example 3: Working with All of the Current Job's Information

```
WRKJOB  OPTION(*ALL)
```

This command allows the user to work with all of the information for the current job.

WRKJOBD (Work with Job Descriptions) Command



Note:
| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Job Descriptions (WRKJOBD) command allows the user to display and work with a list of job descriptions. Job descriptions describe how the system processes jobs.

Restrictions:

1. The user must have *USE authority for the desired job description before the list of job descriptions is displayed.
2. The user must have object operational authority to the job description and use authority to the library in which the job description is located.

Required Parameters

JOBD

Specifies the qualified name of the job description that is shown.

The name of the job description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All job descriptions in the specified library are shown.

job-description-name: Specify the name of the job description.

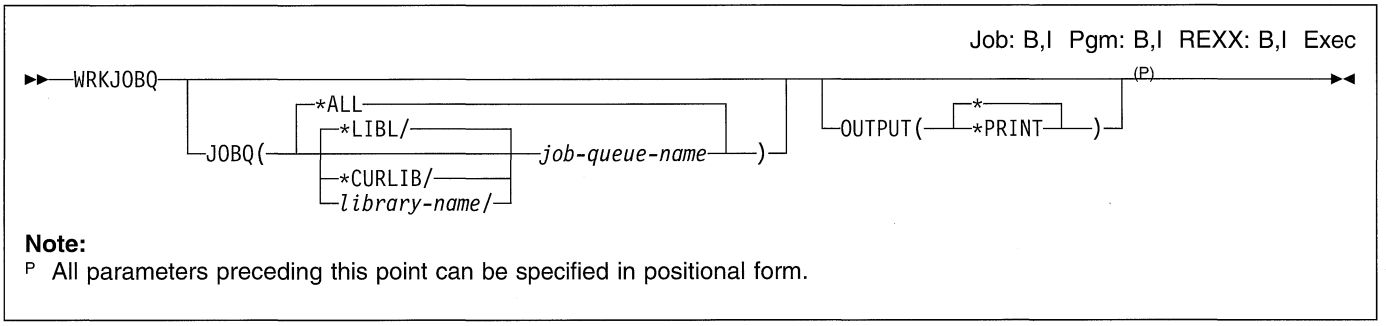
generic-job-description-name:* Specify the generic name of the job description. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Example

```
WRKJOBD  JOBD(MYLIB/SPEC*)
```

This command shows a list of job description names beginning with SPEC in library MYLIB. Options may be selected from this display to work with the job descriptions that are listed.

WRKJOBQ (Work with Job Queue) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Job Queue (WRKJOBQ) command allows the user to work with either the overall status of all job queues or the detailed status of a specific job queue. The status of the queues may change while the command is being run.

Optional Parameters

JOBQ

Specifies that all job queues are shown, or specifies the qualified name of the job queue whose status is shown.

***ALL:** The overall status of all job queues is shown.

| The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-queue-name: Specify the name of the job queue for which detailed status information is shown.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

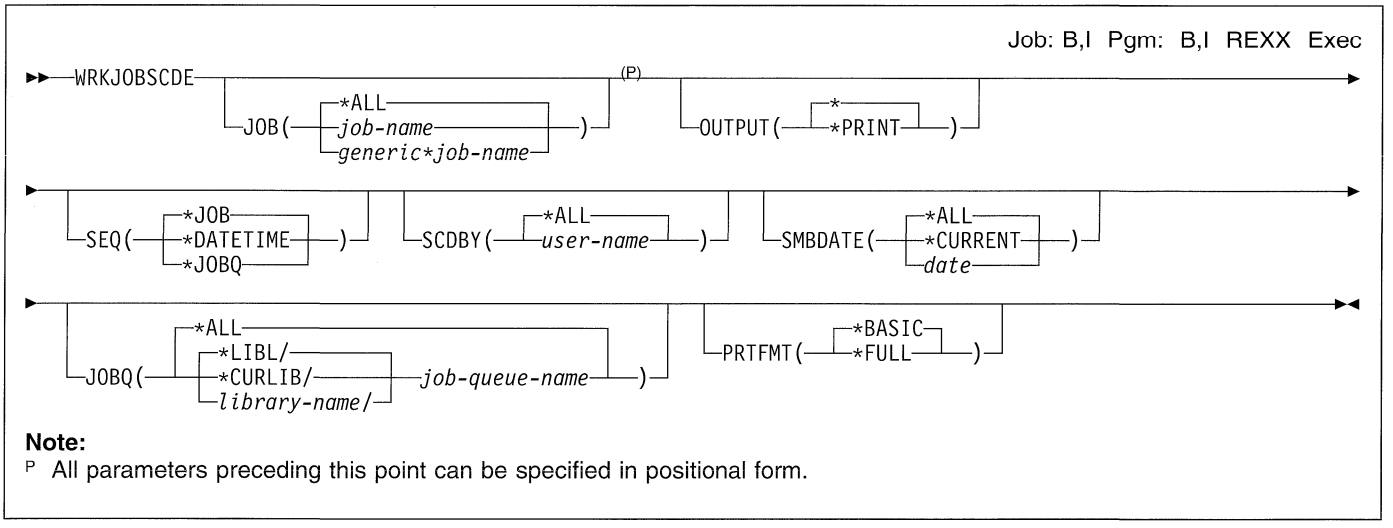
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKJOBQ  JOBQ(QGPL/QBATCH)
```

This command shows the detailed status information about the job queue named QBATCH in the QGPL library. Each job on the QBATCH job queue is identified by job name, user name, and job number; the job's priority and status are also shown.

WRKJOBSCDE (Work with Job Schedule Entries) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Job Schedule Entries (WRKJOBSCDE) command allows you to work with an entry, entries, or generic entries in the job schedule. Each job schedule entry contains the information needed to automatically submit a batch job one time, or at regularly scheduled intervals.

This command shows the Work with Job Schedule Entries display. From the display, you can select options to add, change, remove, hold, or release entries. You can display details of an entry, or work with the last job submitted for an entry. You can also select an option to immediately submit a job using the information contained in a job schedule entry.

Optional Parameters

JOB

Specifies the job name of the job schedule entries with which you want to work.

***ALL:** All job schedule entries that meet this command's other parameter values are shown on the display.

job-name: Specify the name of the job schedule entries to be shown on the display.

| *generic*-job-name:* Specify a generic name. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

SEQ

Specifies the order in which the selected entries are shown.

***JOB:** Entries are shown in alphabetical order by job name. Within a job name the entries are ordered by entry number from lowest to highest.

***DATETIME:** Entries are ordered by the date and time at which their jobs are scheduled to be submitted, with the earliest entries shown first. Entries that do not have a job scheduled to be submitted, such as saved entries, are shown last.

***JOBQ:** The entries are grouped under the name of the job queue to which their jobs are submitted. The job queues are shown in alphabetical order. Within a job queue, entries are shown in alphabetical order.

SCDBY

Specifies the name of the user who added the entry to be shown.

***ALL:** Entries added by all users are shown.

user-name: Specify the name of the user who added the entries to be shown.

SBMDATE
Specifies the date on which the entries to be shown submit jobs to run.

***ALL:** All job schedule entries are shown, regardless of the date on which the entries are to submit a job to run.

***CURRENT:** The entries scheduled to submit jobs on the current date are shown.

date: Specify the submit date for the entries to be shown.

JOBQ
Specifies the name of the job queue to which the jobs are to be submitted. Use this parameter to display all entries that will submit jobs to a specified job queue.

***ALL:** All entries are shown, regardless of the job queues.

The name of the job queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

job-queue-name: Specify the name of the job queue.

PRTFMT
Specifies the format used for the printed output.

***BASIC:** The entries are printed in an abbreviated list format.

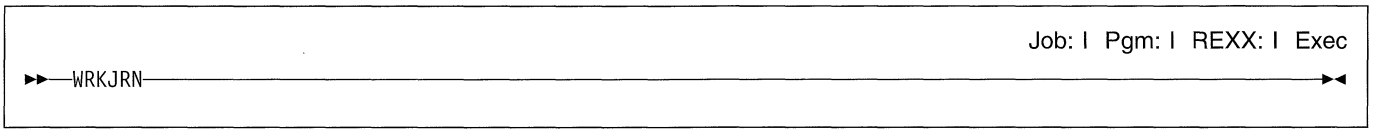
***FULL:** The details of each entry are printed in an expanded format.

Example

```
WRKJOBSCDE  JOBQ(QGPL/QBATCH)
```

This command shows all the job schedule entries that submit a job to the job queue QBATCH in library QGPL.

WRKJRN (Work with Journal) Command



Purpose

The Work with Journal (WRKJRN) command allows you to display and work with a menu from which options for journal operations can be selected. Options on the menu include:

- Display the status of the journal.
- Do forward and back-out file recovery.
- Recover damaged journals and journal receivers.
- Associate journal receivers with a journal.

See the *Advanced Backup and Recovery Guide* for details on the specific options.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, and QSRV user profiles have private authorities to use the command.

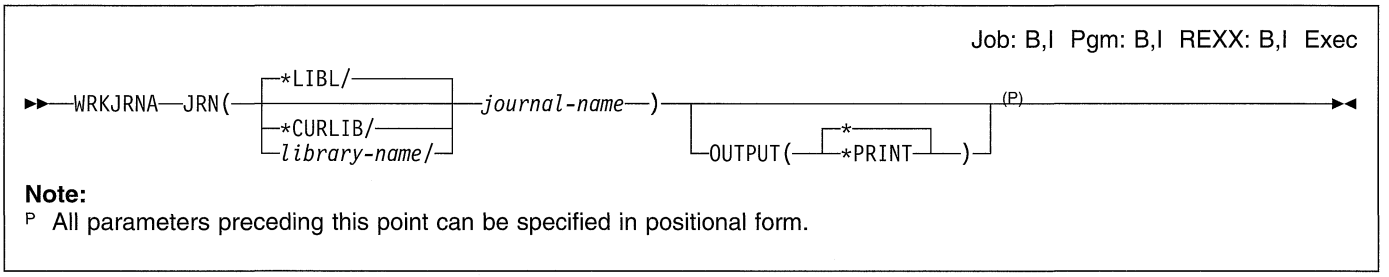
There are no parameters for this command.

Example

```
WRKJRN
```

This command allows the you to utilize the Work with Journal menu.

WRKJRNA (Work with Journal Attributes) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Journal Attributes (WRKJRNA) command allows the user to display and work with the creation and operational attributes of a journal, including the names of the journal receivers currently attached to the journal. From the primary display, options can be selected to display the names of physical files currently journaled, the names of all database files currently having their access paths journaled to the journal, the receiver directory, and detailed information about a journal receiver. Journal receivers can also be deleted from the receiver directory.

If output is printed with the job's spooled printer output, then all of the information that is optionally displayed is printed (except for detailed information about journal receivers; for that information, use the Display Journal Receiver Attributes (DSPJRNRCVA) command).

Required Parameter

JRN

Specifies the qualified name of the journal whose attributes are to be displayed.

The name of the journal can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

journal-name: Specify the name of the journal whose attributes are to be displayed.

Optional Parameter

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

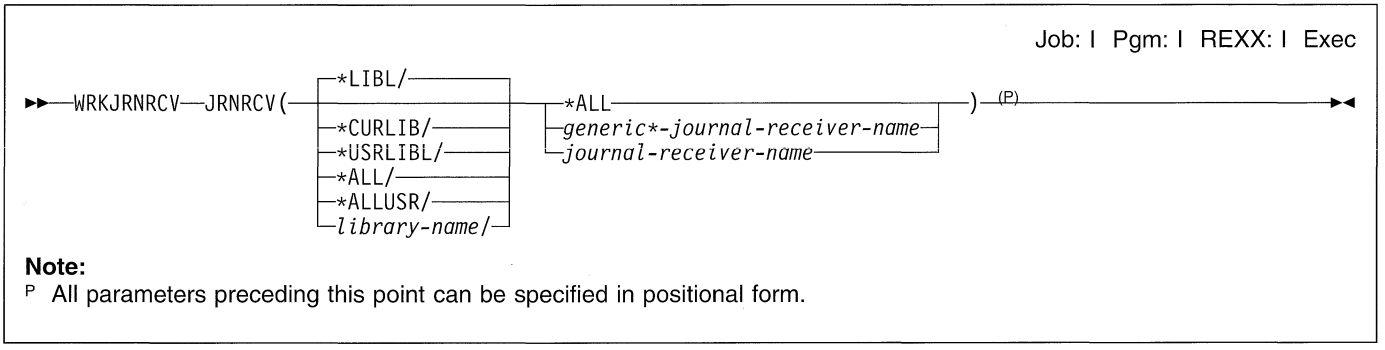
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKJRNA  JRN(MYLIB/JRNLA)
```

This command allows the user to work with the current journal attributes of JRNLA in library MYLIB.

WRKJRNRCV (Work with Journal Receivers) Command



Note:
| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Journal Receivers (WRKJRNRCV) command allows the user to display and work with a list of journal receivers from one or more libraries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the journal receivers to which you have some authority will be shown on the display.
3. To perform operations on the journal receivers, you must have USE authority to the command used by the operation, and the appropriate authority to the journal receivers on which the operation is to be performed.

Required Parameters

JRNRCV

Specifies a list of journal receivers in the libraries that are shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the journal receivers.

| The name of the journal receiver can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All journal receivers in the libraries identified in the library qualifier are shown.

generic-journal-receiver-name:* Specify the generic name of the journal receiver. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be searched only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

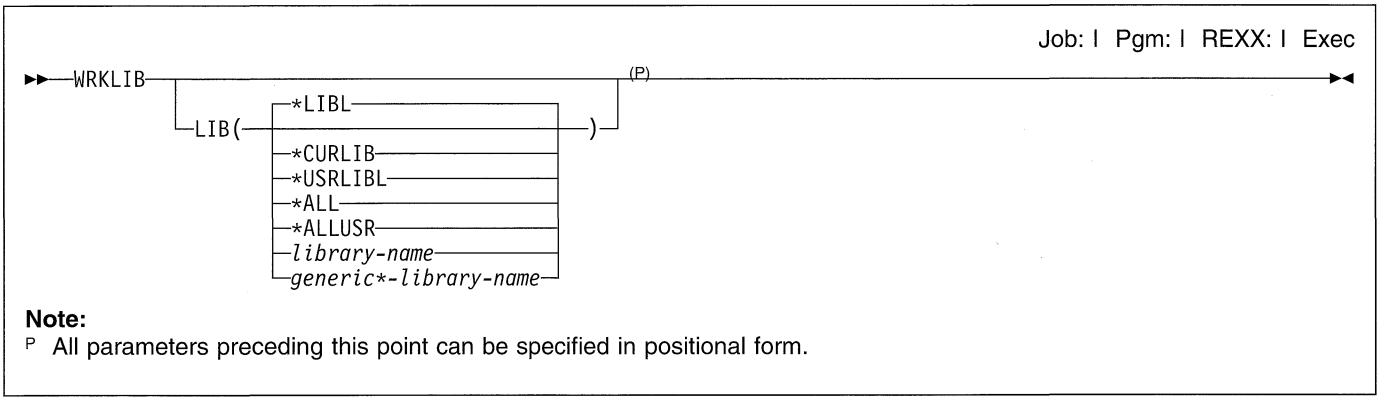
journal-receiver-name: Specify the name of the journal receiver that is shown.

Example

```
WRKJRNRCV  JRNRCV(LIB01/ABC*)
```

This command allows you to display and work with a list of journal receivers beginning with ABC stored in library LIB01.

WRKLIB (Work with Libraries) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Libraries (WRKLIB) command allows you to show and work with a list of names of libraries to which you have authority. From that list several library update functions can be performed.

Restrictions:

1. Only the libraries to which you have some authority will be shown on the display.
2. To perform operations on the libraries, you must have USE authority to the command used by the operation, and the appropriate authority to the libraries on which the operation is to be performed.

Optional Parameters

LIB

Specifies the names of the libraries that are shown or printed.

| The name of the library can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

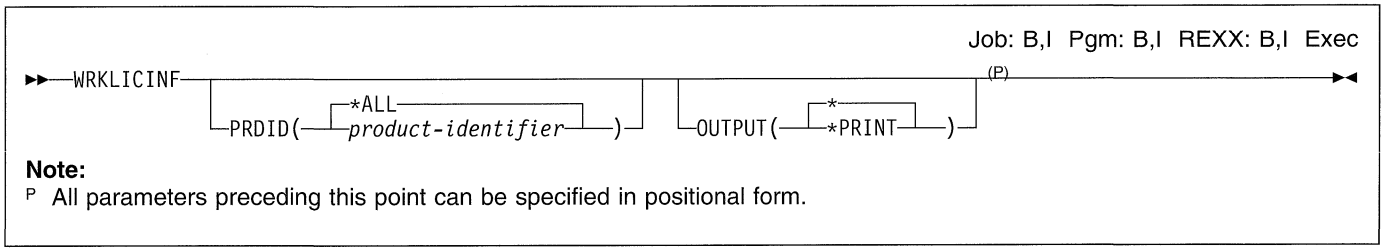
| *generic*-library-name:* Specify the generic name of the library. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Example

```
WRKLIB  LIB(QJ*)
```

This command allows you to work with a list of libraries that begin with the prefix letters QJ.

WRKLICINF (Work with License Information) Command



Purpose

The Work with License Information (WRKLICINF) command allows you to show or print specified products or features found on the system which contain license information. When no parameters are specified, a list of all products with license information is shown. This list allows you to change, display, or print the license information, to reset the peak usage information, or to work with the license users of a product or feature.

Restriction: This command is shipped with public *EXCLUDE authority.

Optional Parameters

PRDID

Specifies the identifier (ID) of the product for which license information is to be displayed.

*ALL: All of the products found on the system which contain license information are displayed.

product-identifier: Specify the seven-character ID of the product for which license information is to be displayed.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

_*:_ The output requested is shown on the display.

***PRINT:** The output is printed with the job's spooled output.

Examples

Example 1: Showing License Information for a Product

```
WRKLICINF  PRDID(1MYPROD)
```

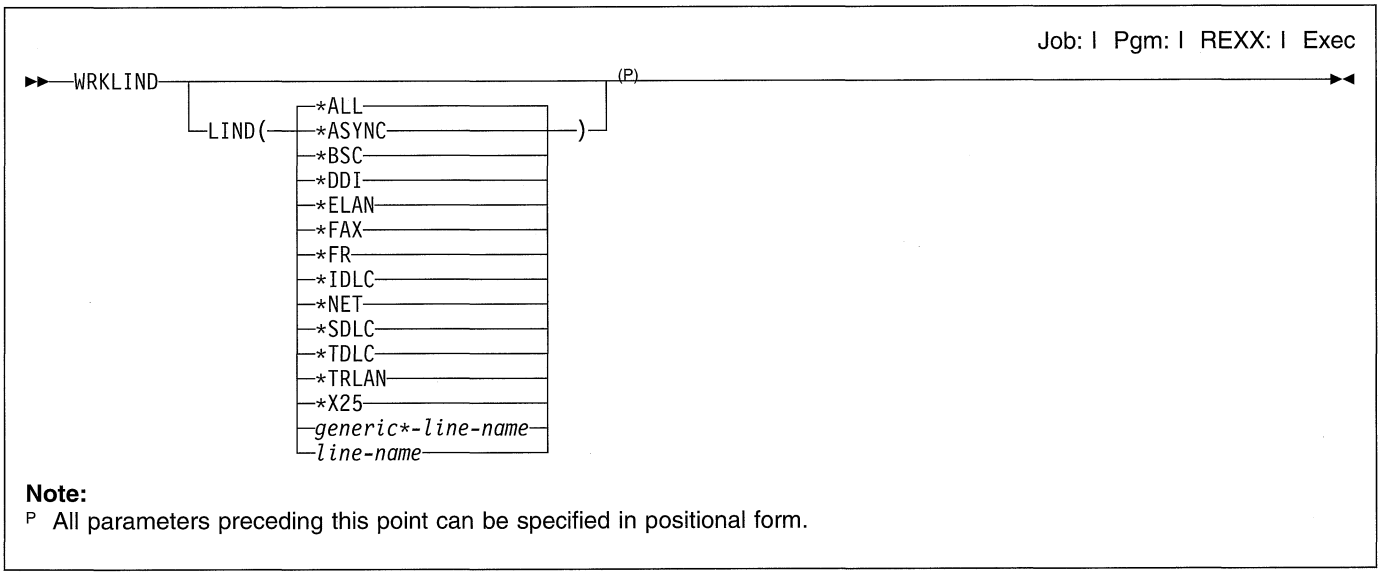
This command allows you to show product license information on your display station for the product with product identifier 1MYPROD.

Example 2: Printing All License Information

```
WRKLICINF  OUTPUT(*PRINT)
```

This command allows you to print, with your job's spooled output, a listing of all (the default value for the PRDID parameter) products on the system which contain license information.

WRKLIND (Work with Line Descriptions) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Line Descriptions (WRKLIND) command allows you to display and work with line description functions by using the Work with Line Descriptions display. This command displays the Work with Line Descriptions display.

Optional Parameters

LIND

Specifies the line descriptions to work with and to include in the list of line descriptions on the Work with Line Descriptions display.

***ALL:** The user can work with all lines.

***ASYNC:** The user can work with all lines configured for asynchronous communications.

***BSC:** The user can work with all lines configured for bisynchronous communications.

***DDI:** The user can work with all lines configured for distributed data interface.

***ELAN:** The user can work with all lines configured for an Ethernet local area network.

***FAX:** The user can work with all lines configured for fax communications.

***FR:** The user can work with all lines configured for frame relay direct communications.

***IDLC:** The user can work with all lines configured for ISDN data link control (IDLC) communications.

***NET:** The user can work with all lines configured for network communications.

***SDLC:** The user can work with all lines configured for synchronous data link control communications.

***TDLC:** The user can work with all lines configured for twinaxial data link communications.

***TRLAN:** The user can work with all lines configured for a token ring local area network.

***X25:** The user can work with all X.25 lines.

generic-line-name:* Specify the generic name of the line. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

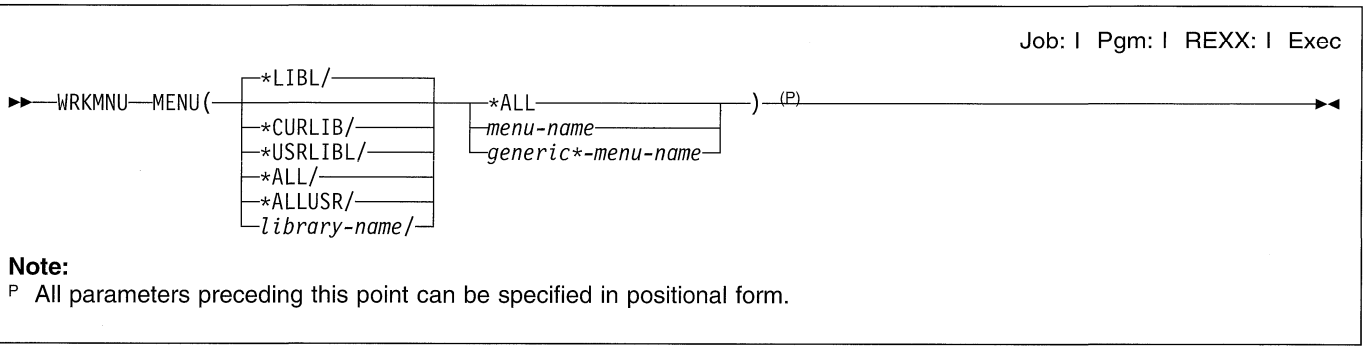
line-name: Specify the name of a specific line description.

Example

```
WRKLIND  LIND(LINE01)
```

This command allows you to work with the Work with Line Descriptions display with an entry for line 'LINE01'. If LINE01 does not exist, the list is blank (no entries are shown).

WRKMNU (Work with Menus) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Menus (WRKMNU) command allows you to display and work with a list of menus and allows the performance of several menu related functions including displaying the menu or menu attributes.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the menus to which you have some authority will be shown on the display.
3. To perform operations on the menus, you must have USE authority to the command used by the operation, and the appropriate authority to the menus on which the operation is to be performed.

Required Parameters

MENU

Specifies the qualified names of the menus shown on the Work with Menus display. A specific menu name or a generic menu name can be specified; either type can be optionally qualified by a library name.

The name of the menu can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All menus specified in the libraries are listed on the Work with Menus display.

menu-name: Specify the name of the menu being listed.

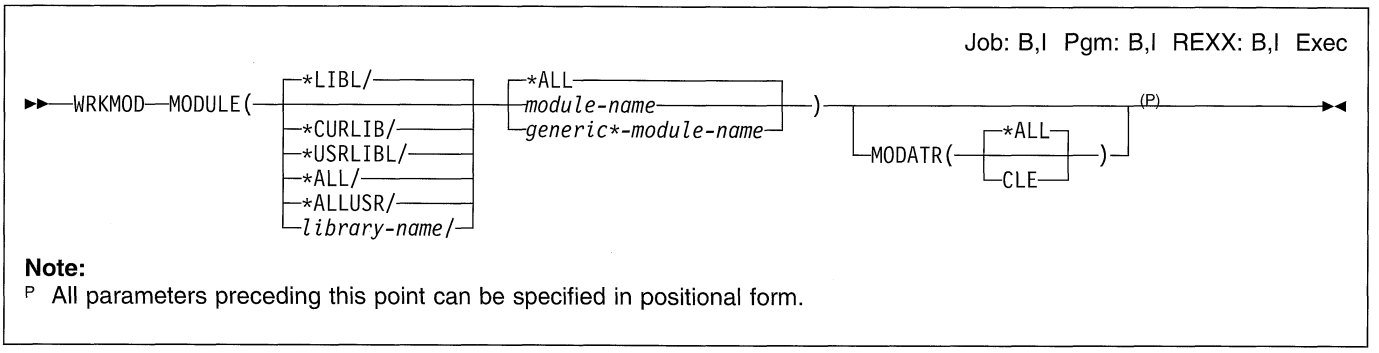
generic-menu-name:* Specify the generic name of the menu. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Example

```
WRKMNU  PERSLIB/OE*
```

This command allows you to work with a list of all menus in library PERSLIB whose names begin with OE.

WRKMOD (Work with Modules) Command



Purpose

The Work with Modules (WRKMOD) command allows you to display and work with a list of modules from one or more libraries.

Restrictions:

1. Only the libraries to which you have *USE authority are searched.
2. Only the modules to which you have authority are shown on the display.
3. To perform operations on the modules, you must have *USE authority to the command used by the operation, and the appropriate authority to the modules on which the operation is to be performed.

Required Parameters

MODULE

Specifies how to search for modules to be placed in the list. All modules with names that correspond to the specified parameter value, and for which the user has authority, are shown.

If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the modules.

The name of the modules can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All modules in the libraries identified in the library qualifier are shown (except those libraries for which the user does not have authority).

module-name: Specify the name of the module shown.

generic-module-name:* Specify the generic name of the module. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all modules with names that begin with the generic name, and for which the user has authority, are shown. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete module name. For more information on the use of generic functions, see "Rules for Specifying Names."

Optional Parameters

MODATR

Specifies that a list of modules with the selected attribute is shown.

***ALL:** Modules are shown regardless of the attribute associated with the module.

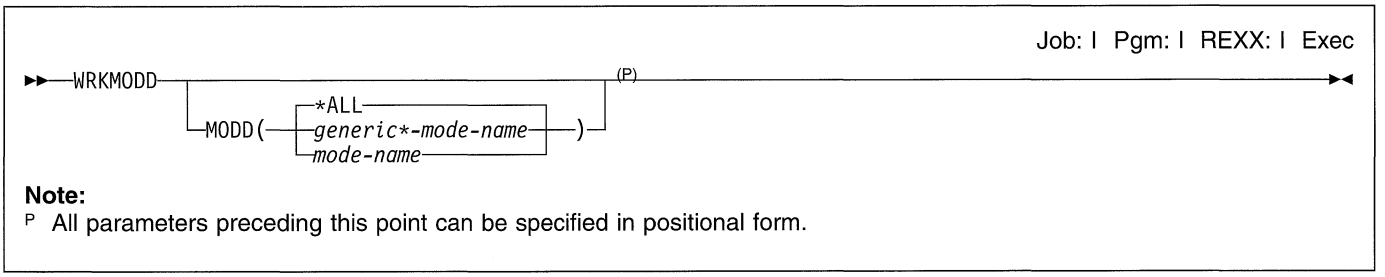
CLE: Modules with the C attribute (C) are shown.

Example

```
| WRKMOD   MODULE(MYLIB/*ALL)
```

| This command lists all the modules to which the user has
| authority that are stored in library MYLIB.

WRKMODD (Work with Mode Descriptions) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Mode Descriptions (WRKMODD) command allows you to display and work with mode description functions by utilizing the Work with Mode Descriptions display.

Optional Parameters

MODD

Specifies the mode descriptions with which you want to work. Specify which mode descriptions to include in the list of mode descriptions on the Work with Mode Descriptions display.

***ALL:** You can work with all mode descriptions.

generic-mode-name:* Specify the generic name of the mode. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

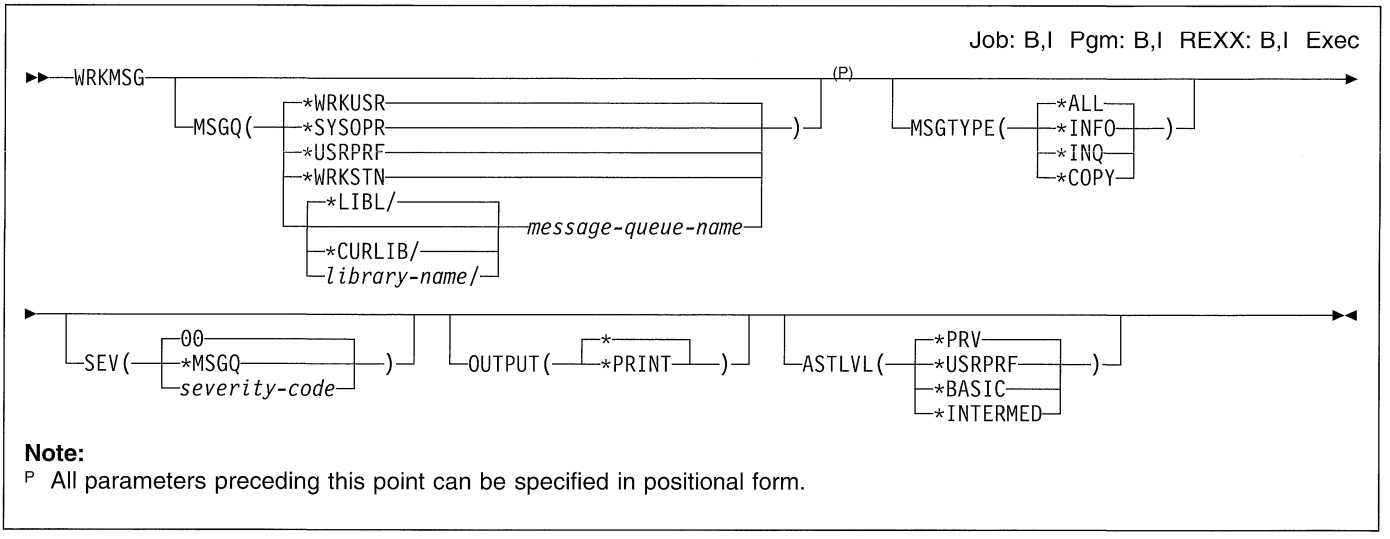
mode-name: Specify a specific mode description.

Example

```
WRKMODD  MODD(*ALL)
```

This command allows you to work with the Work with Mode Descriptions display with entries for all existing mode descriptions.

WRKMSG (Work with Messages) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Messages (WRKMSG) command is used to work with messages received at a specified message queue.

Optional Parameters

MSGQ

Specifies the qualified name of the message queue from which messages are displayed.

***WRKUSR:** Messages from the work station's message queue and the user profile message queue are displayed.

***SYSOPR:** Messages from the system operator message queue (QSYSOPR) are displayed.

***USRPRF:** Messages from the current user profile message queue are displayed.

***WRKSTN:** Messages from the work station message queue are displayed.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

message-queue-name: Specify the qualified name of the message queue from which messages are shown.

MSGTYPE

Specifies the type of messages in the queue displayed.

***ALL:** All messages in the queue are displayed.

***INFO:** Informational messages (those not requiring a reply) are displayed.

***INQ:** Inquiry messages (those requiring a reply) are displayed.

***COPY:** The sender's copy of an inquiry message that was sent to a queue and required a reply is displayed.

SEV

Specifies the severity code of the message. The severity code indicates the severity level of the condition that causes the message to be sent.

00: All messages in the specified queue are displayed.

***MSGQ:** All messages having a severity code value greater than or equal to the severity code specified for the message queue are shown.

severity-code: Specify the lowest severity code value that a message can have and still be displayed. Valid values range from 00 through 99.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output. Immediate messages and predefined messages are truncated to 105 characters when printed.

***PRINT:** The output is printed with the job's spooled output.

ASTLVL

Specifies the user interface to display.

***PRV:** The previous user interface used is displayed.

***USRPRF:** The user interface specified in the current user profile is displayed.

***BASIC:** The Work with Messages display is shown. This user interface separates messages into two categories: 1) messages needing a reply, and 2) messages not needing a reply. New messages are shown at the top of each message list.

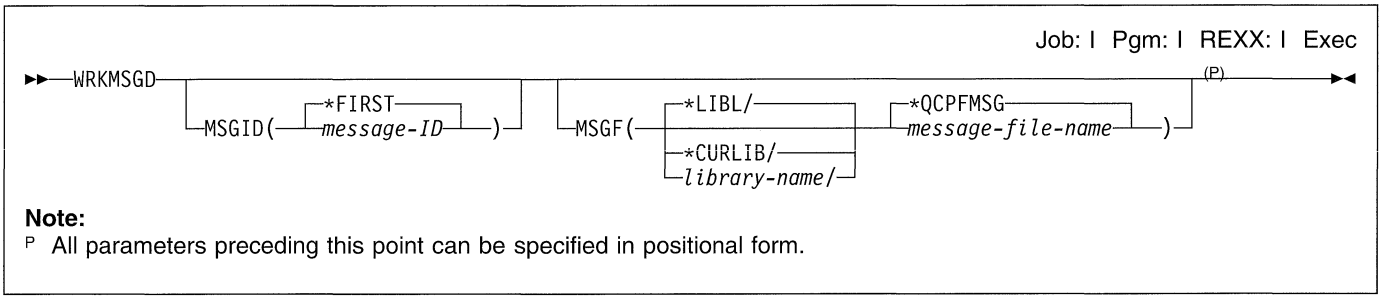
***INTERMED:** The Display Messages display is shown.

Example

```
WRKMSG
```

This command displays all messages from the requester's work station message queue and user profile message queue. Messages needing a reply are displayed first, followed by messages not needing a reply. Messages are displayed from newest to oldest.

WRKMSGD (Work with Message Descriptions) Command



Purpose

The Work with Message Descriptions (WRKMSGD) command allows the user to display and work with detailed information about the messages contained in a message file. This command is used to add, change, display, print, and remove message descriptions. When message descriptions are changed by using this display, the current values for the message are shown in the command prompt. However, there is a 512-character limit for the second-level message text.

Optional Parameters

MSGID

Specifies the message identifier with which to begin showing a list of message descriptions in the message file specified by the MSGF parameter.

***FIRST:** The first message description in the message file is first in the message list display.

message-ID: Specify the message identifier used to begin the message list display. The message identifier must be seven characters in length and in the format, `pppmmnn`.

The first three characters (ppp) must begin with a code consisting of an alphabetic character followed by two alphanumeric (alphabetic or decimal) characters; the last four characters (mmnn) must consist of decimal values ranging from 0 through 9 and the characters A through F.

MSGF

Specifies the qualified name of the message file from which the message descriptions are taken.

The name of the message file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

***QCPFMSG:** Message descriptions are taken from the system message file, QCPFMSG, in library QSYS.

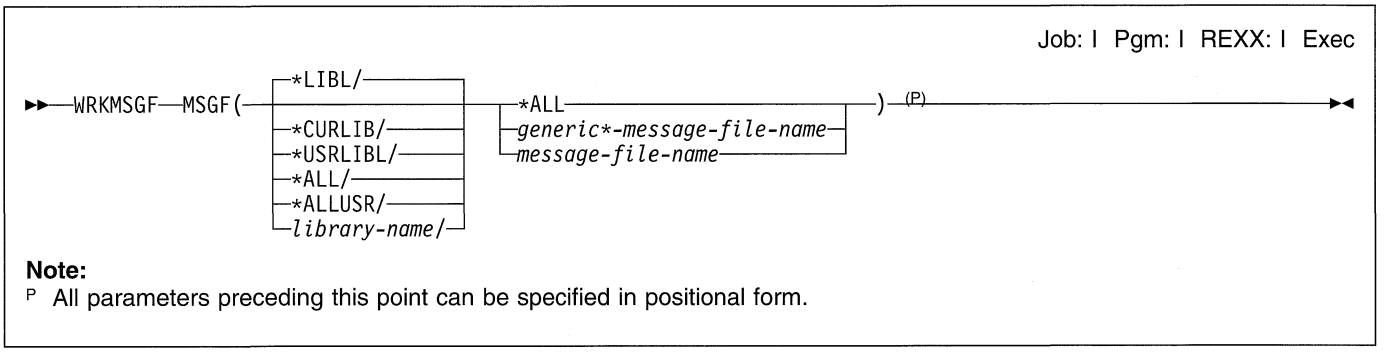
message-file-name: Specify the name of the message file to use.

Example

```
WRKMSGD  MSGF(QSYS/QCPFMSG)
```

This command allows the user to utilize the Work with Message Descriptions display for message descriptions found in message file QCPFMSG in library QSYS. From that display, the user can add, change, delete, display, or print these message descriptions.

WRKMSGF (Work with Message Files) Command



Note:
(P) All parameters preceding this point can be specified in positional form.

Purpose

The Work with Message Files (WRKMSGF) command allows you to display and work with a list of message files from one or more libraries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the message files to which you have some authority will be shown on the display.
3. To perform operations on the message files, you must have USE authority to the command used by the operation, and the appropriate authority to the message files on which the operation is to be performed.
4. You must have object operational authority for the message file.

Required Parameters

MSGF

Specifies a list of message files in the libraries that are shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the library list are searched for the message files.

The name of the message file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All message files in the libraries identified in the library qualifier are shown.

generic-message-file-name:* Specify the generic name of the message file. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

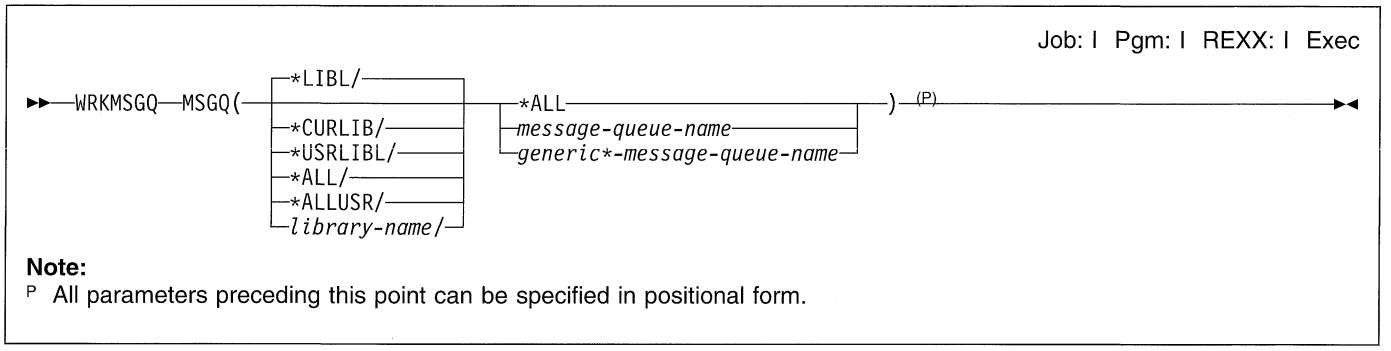
message-file-name: Specify the name of the message file to use.

Example

```
WRKMSGF  MSGF(ACCNTLIB/*ALL)
```

This command lists all the message files in the ACCNTLIB library.

WRKMSGQ (Work with Message Queues) Command



Note:
(P) All parameters preceding this point can be specified in positional form.

Purpose

The Work with Message Queues (WRKMSGQ) command allows you to display and work with a list of message queues and allows you to display, change, delete, and clear specified message queues.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the message queues to which you have some authority will be shown on the display.
3. To perform operations on the message queues, you must have USE authority to the command used by the operation, and the appropriate authority to the message queues on which the operation is to be performed.

Required Parameters

MSGQ

Specifies the qualified name of the message queues being shown on the Work with Message Queues display. A specific message queue, or a generic message queue, can be specified; either type can be optionally qualified by a library name.

The name of the message queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All message queues (to which the user has some authority) in the specified library are listed on the Work with Message Queues display.

message-queue-name: Specify the name of the message queue being listed.

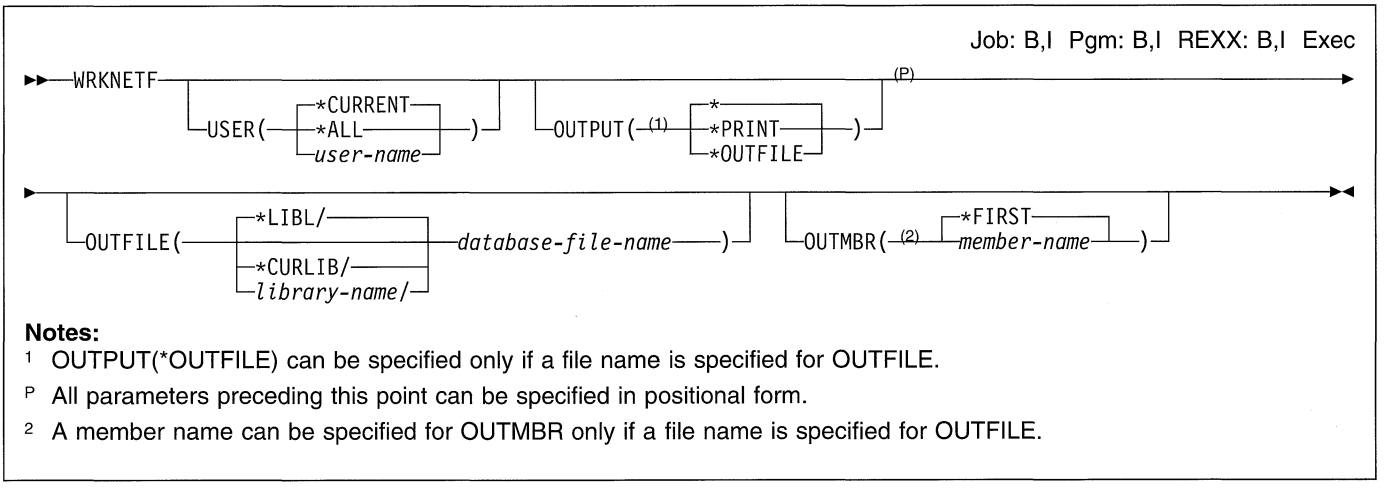
generic-message-queue-name:* Specify the generic name of the message queue. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Example

```
WRKMSGQ  PERSLIB/MQ*
```

This command allows you to work with a list of all message queues whose names begin with MQ in library PERSLIB.

WRKNETF (Work with Network Files) Command



Notes:
1 OUTPUT(*OUTFILE) can be specified only if a file name is specified for OUTFILE.
P All parameters preceding this point can be specified in positional form.
2 A member name can be specified for OUTMBR only if a file name is specified for OUTFILE.

Purpose

The Work with Network Files (WRKNETF) command allows you to work with a list of files that have arrived for a user, or creates an output file containing a list of the files.

If the list is shown, you can enter an option to select a function to be performed on the file. You can:

- Receive the file into a user file.
- Delete the file.
- Browse the file (not valid for save files).
- Submit files (submit the input stream).

Restrictions:

1. A user with security officer authority can display the network files for any user. Users other than the security officer can show only those files that were sent to them or to their group profile.
2. To perform any of the options from this display, you must be authorized to the command corresponding to that option. For example, you must be authorized to the Display Physical File Member (DSPPFM) command for the browse function, and the Submit Database Jobs (SBMDBJOB) command for the submit job function.

Optional Parameters

USER
Specifies the user for whom the files are shown.

***CURRENT:** The user profile under which the current job is running is used.

***ALL:** The network files for all users are shown. Only a user with security officer authority can specify this value.

user-name: Specify the name of the specified user whose files are shown. Only users with security officer authority can specify a name other than that of their own or group profiles.

OUTPUT
Specifies whether the output from the command is shown at the requesting work station, listed with the job's spooled output, or directed to a database file. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

***OUTFILE:** The output is directed to the database file specified on the OUTFILE parameter.

OUTFILE
Specifies the name of the database file to which the output of the display is directed. If the specified data base file does not exist, this command creates it in the specified library.

Note: The outfile format must be the same as QNFDNTF of the system file QANFDNTF.

The name of the database file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

database-file-name: Specify the name of the database file that receives the output of the display.

OUTMBR
Specifies the name of the database file member to which the output is directed.

***FIRST:** The first member in the file receives the output. If OUTMBR(*FIRST) is specified and the member does not exist, the system creates a member with the name of the file specified on the OUTFILE parameter.

member-name: Specify the file member that receives the output. If OUTMBR(member-name) is specified and the member does not exist, the system creates it.

Examples

Example 1: Working with User's Network Files

```
WRKNETF
```

This command allows you to work with all network files for the user running this command. If the command is issued as an interactive job, the list of files is shown at the requesting work station. If the command is issued as a batch job, the list of files is printed with the job's spooled output.

Example 2: Printing Output

```
WRKNETF  USER(USR1)  OUTPUT(*PRINT)
```

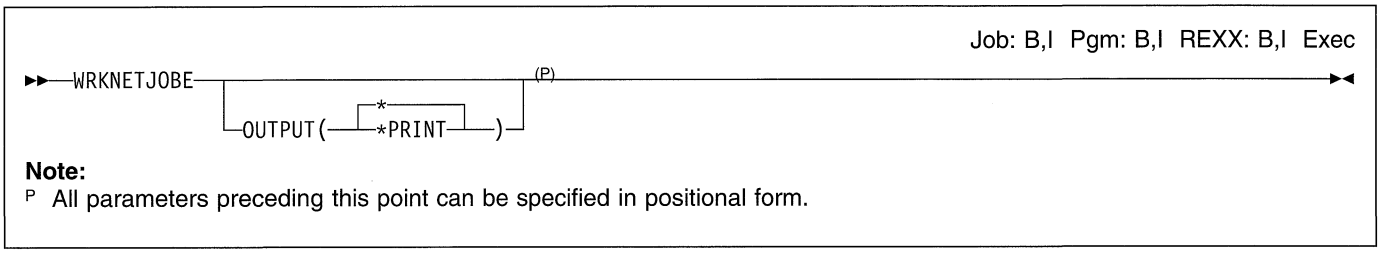
This command allows you to work with the network files for USR1 and prints the output with the job's spooled output. This command can only be issued by USR1, a member of the USR1 group, or a user with security officer authority.

Example 3: Working with Network Files for All Users

```
WRKNETF  USER(*ALL)  OUTPUT(*OUTFILE)
  OUTFILE(NETFILES)
```

This command allows you to work with the network files for all users and is written to the first member of a database named NETFILES. If the file exists in a library on the library list, the existing file is used; otherwise, the file is created in the QGPL library. If the file did not exist, or did not contain any members, a member with the same name as the file is added to the file; otherwise, the first member of the file is cleared and used. This command can be issued only by a user with security officer rights.

WRKNETJOBE (Work with Network Job Entries) Command



Note:
| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Network Job Entry (WRKNETJOBE) command allows you to work with the network job entries. There is one entry for each user or distribution group who may submit jobs to this system. This entry is used to determine whether the input stream is automatically submitted, placed on the queue of network files for a user, or rejected. This entry also specifies the user profile that is used for checking the authority to the job description referenced in the input stream.

Restrictions:

1. Any user can display the network job entries with this command.
2. You must be explicitly authorized to the commands corresponding to the options that can be selected from this display in order to select these options.

Optional Parameters

OUTPUT

Specifies whether the output is shown at the requesting work station or printed with the job's spooled output.

| More information on this parameter is in Appendix A,
| "Expanded Parameter Descriptions."

*: Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

Examples

Example 1: Printing Output

```
WRKNETJOBE  OUTPUT(*PRINT)
```

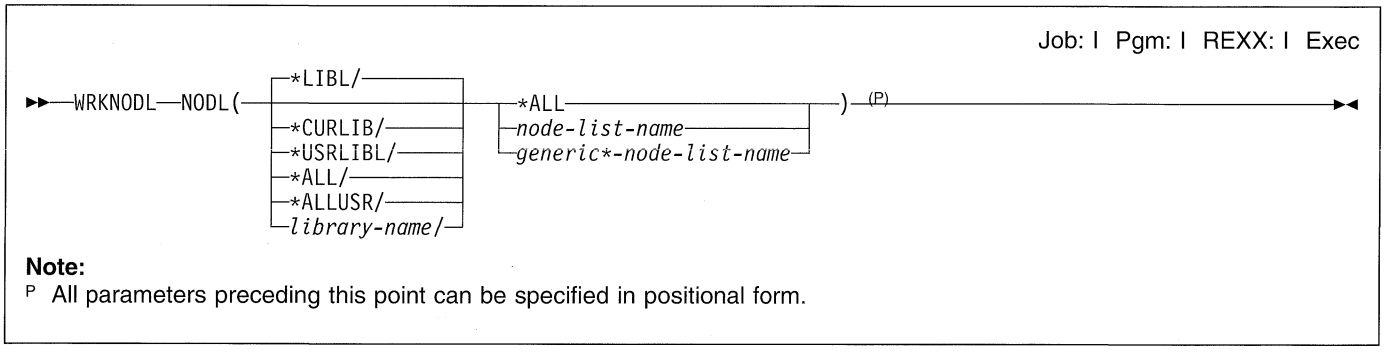
This command allows you to work with the network job entries that are printed with the job's spooled output.

Example 2: Working with Network Job Entries

```
WRKNETJOBE  OUTPUT(*)
```

This command, if issued in an interactive job, allows you to work the network job entries at the requesting work station. If the command is issued in a batch job, the network job entries are printed with the job's spooled output.

WRKNODL (Work with Node Lists) Command



Purpose

The Work Node List (WRKNODL) command allows the user to work with a list of nodes list objects, to create a new node list, delete existing node lists, and to work with node list entries.

Restrictions:

1. Only the libraries to which you have *USE authority are searched.
2. Only the node list to which you have some authority is shown on the display.
3. To perform operations on the node lists, you must have *USE authority to the command used by the operation, and the appropriate authority to the node list on which the operation is to be performed.

Required Parameters

NODL

Specifies the qualified name of the node lists that are shown.

The name of the node list can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All the node lists specified in the library are listed.

node-list-name: Specify the name of the node list that is shown.

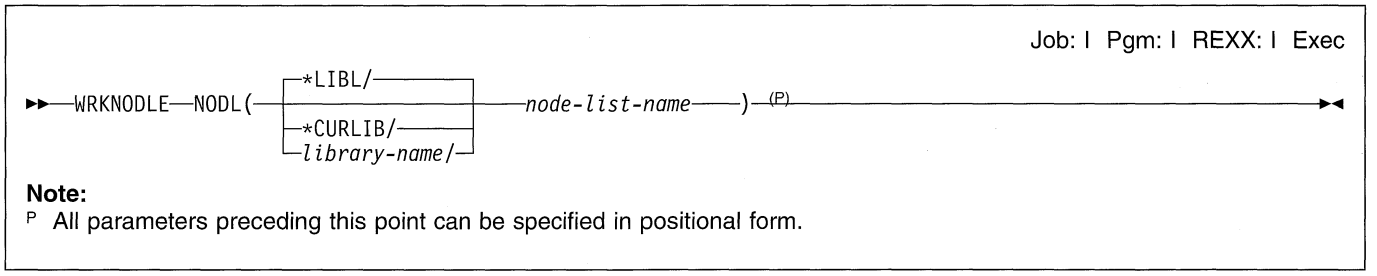
generic-node-list-name:* Specify the generic name of the node list. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Example

```
WRKNODL   NODL(MYLIB/MY*)
```

This command shows a list of all node lists in library MYLIB whose names begin with MY.

WRKNODLE (Work with Node List Entries) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Node List Entries (WRKNODLE) command allows the user to display, print, add, or remove node list entries.

Required Parameters

NODL

Specifies the qualified name of the node list object from which entries are shown.

The name of the node list object can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

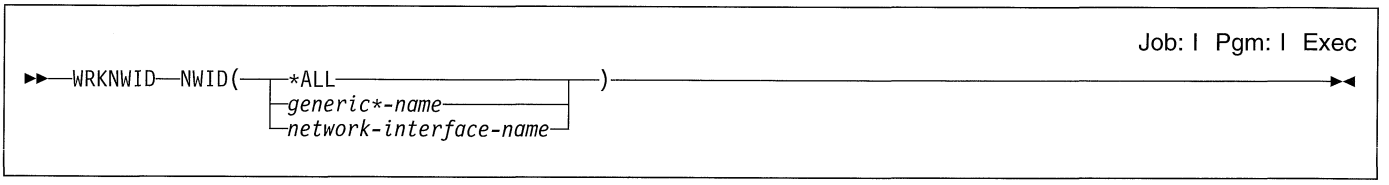
node-list-name: Specify the name of the node list to use.

Example

```
WRKNODLE   NODL(MYLIB/NODL02)
```

This command shows a list of all entries in the node list NODL02 in library MYLIB.

WRKNWID (Work with Network Interface Description) Command



Purpose

The Work with Network Interface Description (WRKNWID) command displays the Work with Network Interface Description menu, which provides an interactive interface to network interface description functions.

Required Parameters

NWID

Specifies the network interface descriptions to work with.

***ALL:** The user can work with all network interfaces.

generic-name:* Specify the generic name of the network interface description. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

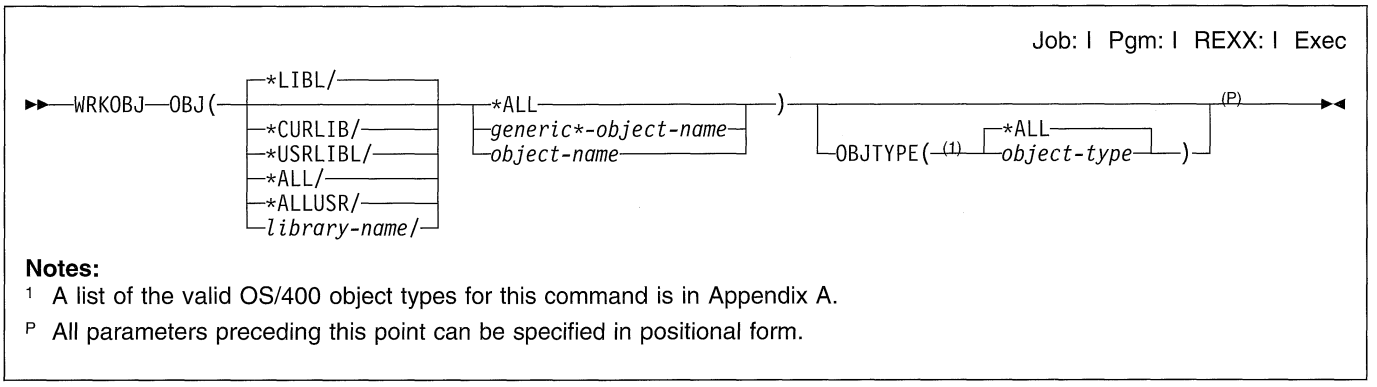
network-interface-name: Specify the name of the network interface to work with.

Example

```
WRKNWID  NWID(NET1)
```

This command displays the Work with Network Interface Descriptions panel with an entry for network interface 'NET1'. If NET1 does not exist, no entries are displayed.

WRKOBJ (Work with Objects) Command



Notes:

1 A list of the valid OS/400 object types for this command is in Appendix A.

P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Objects (WRKOBJ) command shows a list of names and attributes of specified objects in specified libraries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the objects to which you have some authority will be shown on the display.
3. To perform operations on the objects, you must have USE authority to the command used by the operation, and the appropriate authority to the objects on which the operation is to be performed.
4. You must have object operational authority to the object.

Required Parameter

OBJ

Specifies which objects in the libraries are shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the objects. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

The name of the object can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All objects in the libraries identified in the library qualifier that are of the types specified by the OBJTYPE parameter are shown.

generic-object-name:* Specify the generic name of the object. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

object-name: Specify the name of the object that is shown. If the library qualifier is *ALL, *ALLUSR, or a library name, all objects of the specified types to which you have some authority (for example, *USE authority), and that are in the specified libraries, are shown.

Optional Parameter

OBJTYPE

Specifies which types of objects are shown. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ALL:** All object types are shown that have the specified object name.

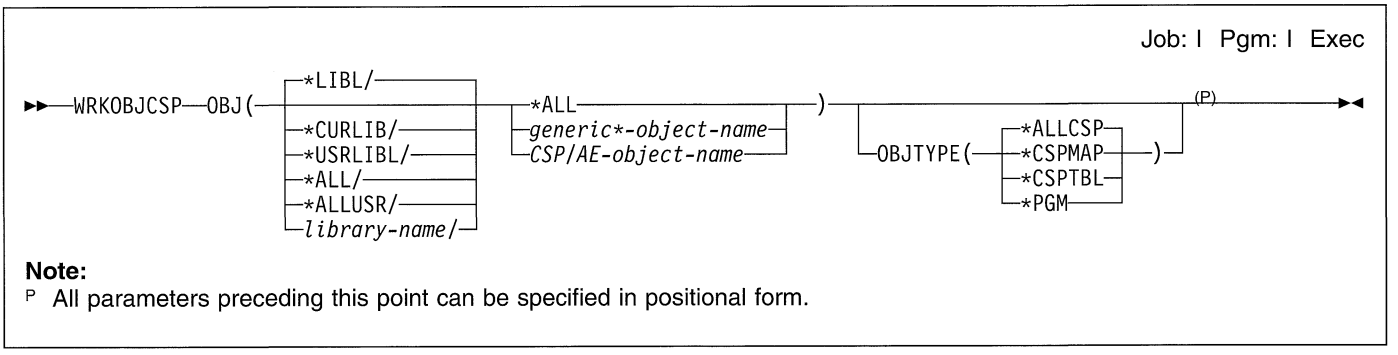
object-type: Specify a value for the types of objects that can be shown.

Example

```
WRKOBJ  OBJ(X/PAY)  OBJTYPE(*ALL)
```

This command permits you to work with the objects for which you have authority that are named PAY and are located in library X.

WRKOBJCSP (Work with Objects for CSP/AE) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Objects for CSP/AE (WRKOBJCSP) command allows you to work with a list of Cross System Product/Application Execution (CSP/AE) application objects. From the selection display shown by this command you can:

- Change a CSP/AE program's commitment lock level value for an SQL application.
- Copy one or more CSP/AE objects to a different library.
- Delete one or more CSP/AE objects.
- Print one or more CSP/AE applications.

  Printing provides a list of the names of AS/400 objects required by the specified CSP/AE application.
- Rename one or more objects in a library.
- Run a CSP/AE application interactively.
- Change the description text associated with a CSP/AE object.

Required Parameters

OBJ

Specifies the qualified name of the CSP/AE objects to appear on the Work with Objects for CSP/AE Applications display.

| The name of the CSP/AE objects can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All objects that are of the object type specified on the OBJTYPE parameter are listed.

generic-object-name:* Specify the generic name of the object. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

CSP/AE-object-name: Specify the name of the object.

Optional Parameters

OBJTYPE

Specifies which types of objects are listed.

***ALLCSP:** All program, map group, and table object types are listed.

***CSPMAP:** Only CSP/AE map groups are listed.

***CSPTBL:** Only CSP/AE tables are listed.

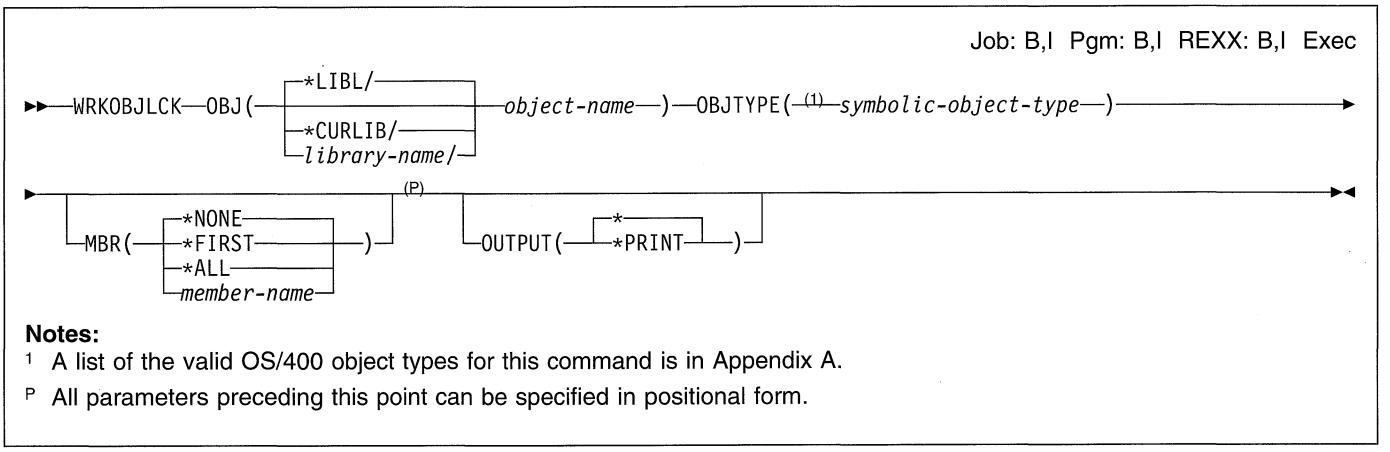
***PGM:** Only CSP/AE programs are listed.

Example

```
WRKOBJCSP  OBJ(CSPLIB/M*)  OBJTYPE(*ALLCSP)
```

This command allows you to work with all CSP/AE programs, map groups, and tables from the library CSPLIB that start with the letter "M."

WRKOBJLCK (Work with Object Locks) Command



Notes:

1 A list of the valid OS/400 object types for this command is in Appendix A.

P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Object Locks (WRKOBJLCK) command allows the user to display and work with or change the object lock requests for a specified object in the system. The user can work with both held locks and locks waiting to be applied.

Restrictions:

1. This command does not show record locks for database files.
2. Work station message queues cannot be allocated, and therefore, they will not have any locks. A work station message queue is associated with a work station device description of the same name. Therefore, to determine why an operation that requires the work station message queue to be allocated does not work, the user should see if there are any locks on the device description of the same name.

Required Parameters

OBJ

Specifies the qualified name of the object for which locks are being displayed.

The name of the object can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

object-name: Specify the name of the object.

If a file member is specified for a file, and the file's library value is *LIBL, the first occurrence of the file in the job's library search list is searched for the member. For object types that exist only in library QSYS (for example, *DEVD), QSYS and *LIBL are the only library names that are accepted.

OBJTYPE

Specifies the object type of the operating system object for which locks are being displayed. Specify the predefined value that identifies the object type. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Optional Parameters

MBR

Specifies the member name of a database file. This parameter is valid only when a database file has been specified on the OBJ parameter.

***NONE:** No member locks are displayed, but file level locks are displayed. The display of member locks for all the members in the file may be requested from the File Locks display.

***FIRST:** The first member in the database file is used.

***ALL:** Member locks for all the members in the file are displayed.

member-name: Specify the name of the database file member for which locks are displayed.

OUTPUT

Specifies whether output from this command is displayed at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

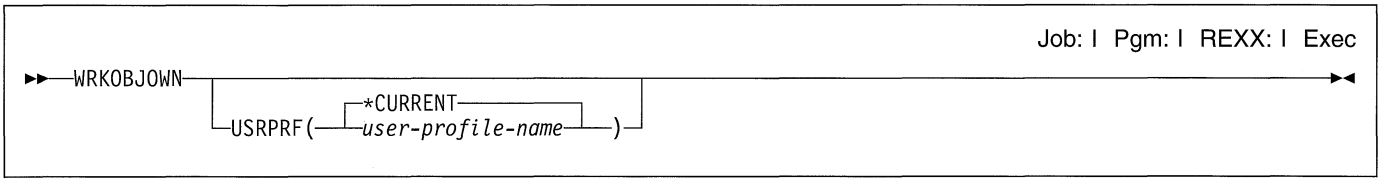
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKOBJLCK  OBJ(*LIBL/LOCKEDFILE)  OBJTYPE(*FILE)
  MBR(LOCKEDMBR)  OUTPUT(*PRINT)
```

This command prints the lock information for the member named LOCKEDMBR in the file named LOCKEDFILE.

WRKOBJOWN (Work with Objects by Owner) Command



Purpose

The Work with Objects by Owner (WRKOBJOWN) command allows you to display and work with objects owned by any user profile. This command allows you to display objects owned by a user profile and perform the following:

- Edit object authority
- Delete the object
- Display object authority
- Display the object's description
- Change ownership of the object

Restriction: The user must have read authority to the profile being displayed.

Optional Parameter

USRPRF

Specifies the name of the user profile whose objects are being worked with.

***CURRENT:** The user profile under which the current job is running is used.

user-profile-name: Specify the name of the user profile being worked with.

Examples

Example 1: Working With Current User Profile

```
WRKOBJOWN  USRPRF(*CURRENT)
```

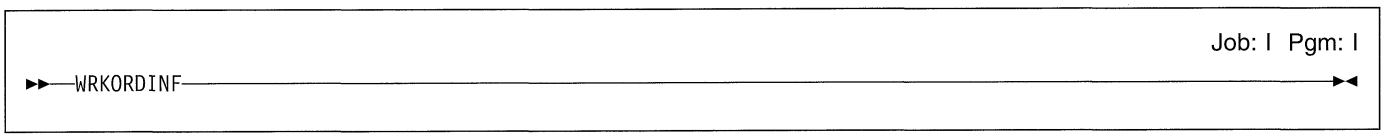
This command allows you to work with the objects owned by the user running this command.

Example 2: Working With a User Profile

```
WRKOBJOWN  USRPRF(BARTH)
```

This command allows you to work with the objects owned by user profile BARTH.

WRKORDINF (Work with Order Information) Command



Purpose

The Work with Order Information (WRKORDINF) command allows you to do the following:

- Automatically create an order information file for the system each time the command is run.
- Copy order information files to or from selected media.
- Send order information files to IBM.

Restrictions:

1. This command is shipped with public *EXCLUDE authority.
2. You must have *ALLOBJ authority or be signed on as QSYSOPR or QSRV to use the command.

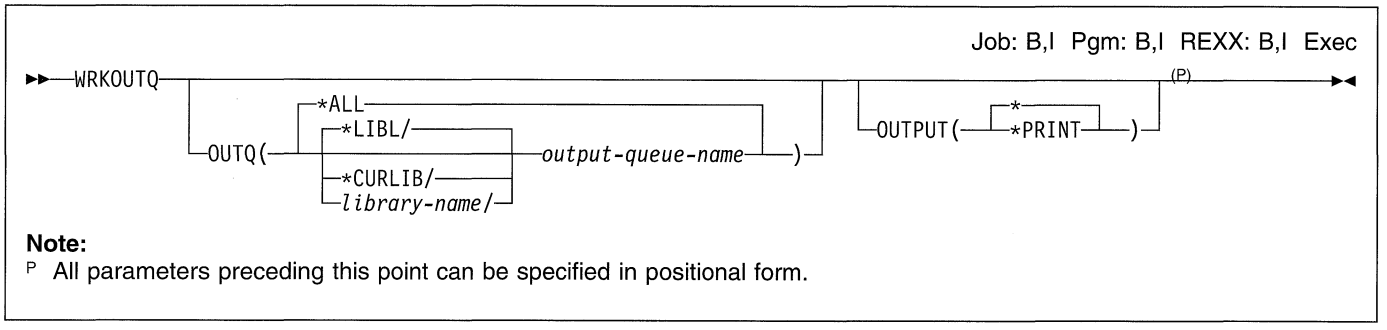
There are no parameters for this command.

Example

```
WRKORDINF
```

This command displays the Work with Order Information menu.

WRKOUTQ (Work with Output Queue) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Output Queue (WRKOUTQ) command allows the user to display and work with either the overall status of all output queues to which the user is authorized, or the detailed status of a specific output queue. The status of the queues may change while the command is being run.

Optional Parameters

OUTQ

Specifies either that the status of all output queues is shown, or specifies the qualified name of a single output queue for which the status is shown.

***ALL:** The overall status of all output queues is shown.

| The name of the output queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

output-queue-name: Specify the name of the output queue whose detailed status information is shown.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or is printed with the job's spooled output. More information on this
| parameter is in Appendix A, "Expanded Parameter
| Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

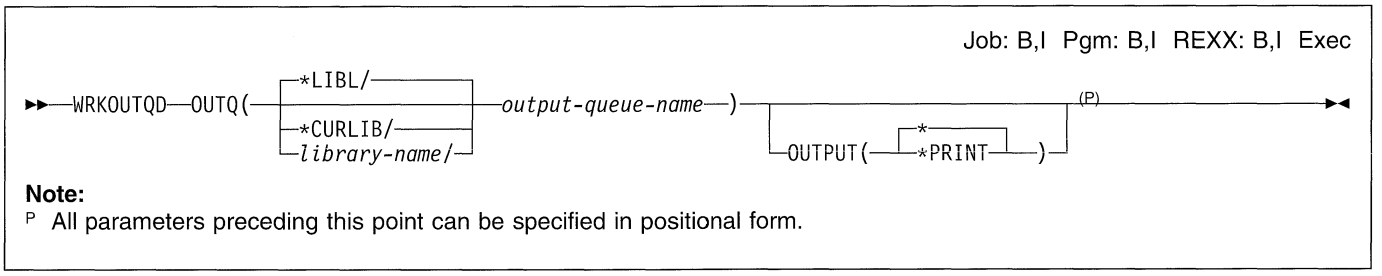
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKOUTQ  OUTQ(QGPL/QPRINT)
```

This command allows the user to show and work with the detailed status information about the output queue named QPRINT in the QGPL library. Each spooled file on the QPRINT output queue is shown.

WRKOUTQD (Work with Output Queue Description) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Required Parameter

The Work with Output Queue Description (WRKOUTQD) command allows the user to work with the description of the specified output queue. The description of the queue may change while the command is being run.

Required Parameters

OUTQ

Specifies the qualified name of the output queue.

| The name of the output queue can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

output-queue-name: Specify the name of the output queue.

Optional Parameter

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output on a printer. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

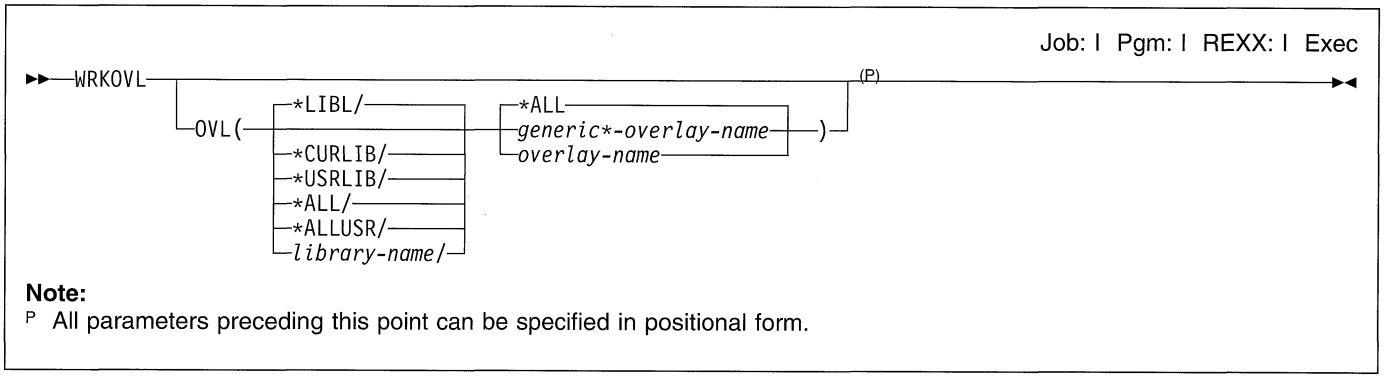
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKOUTQD  OUTQ(QGPL/QPRINT)
```

This command allows the user to work with the descriptive information for the output queue named QPRINT which is in the QGPL library.

WRKOVL (Work with Overlays) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Overlays (WRKOVL) command allows you to display and work with all of the overlays from the system or user libraries (or both).

Optional Parameters

OVL

Specifies the qualified name of the overlay.

The name of the overlay can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All overlays in the libraries identified in the library qualifier are shown. Only those overlays for which the user has some authority are shown.

generic-overlay-name:* Specify the generic name of the overlay. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

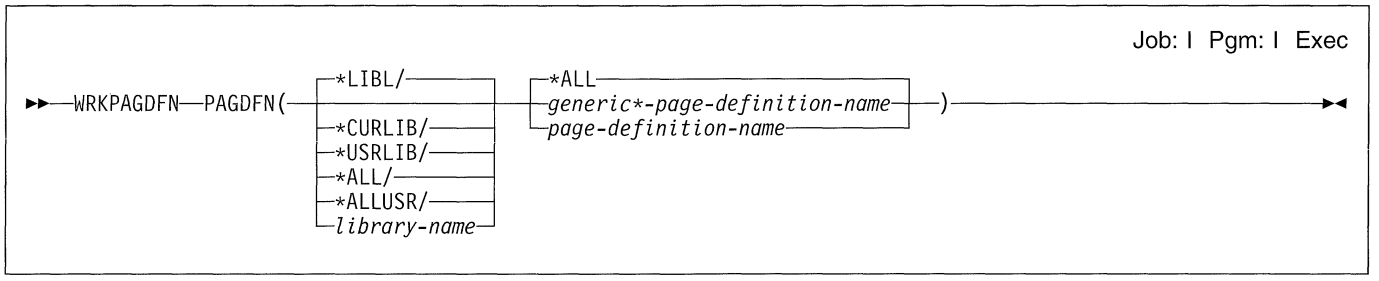
overlay-name: Specify the name of the overlay to be listed.

Example

```
WRKOVL  OVL(*LIBL/OV*)
```

This command searches the library list to find the overlays whose names begin with OV. All overlay objects with names beginning with OV are shown on the display. If no overlay objects beginning with OV exist in the library list, the WRKOVL display is shown with a message indicating that an object matching the specified name cannot be found.

WRKPAGDFN (Work with Page Definitions) Command



Purpose

The Work With Page Definitions (WRKPAGDFN) command displays a list of page definitions. From the display, you can create or delete a page definition, display the name and attributes of a page definition, or change the description of a page definition.

Required Parameters

PAGDFN

Specifies the qualified name of the page definition to be listed on the Work with Page Definitions display. Only those page definitions for which the user has *READ authority are shown.

The name of the page definition can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All page definitions are listed.

generic-page-definition-name:* Specify the generic name of the page definition. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

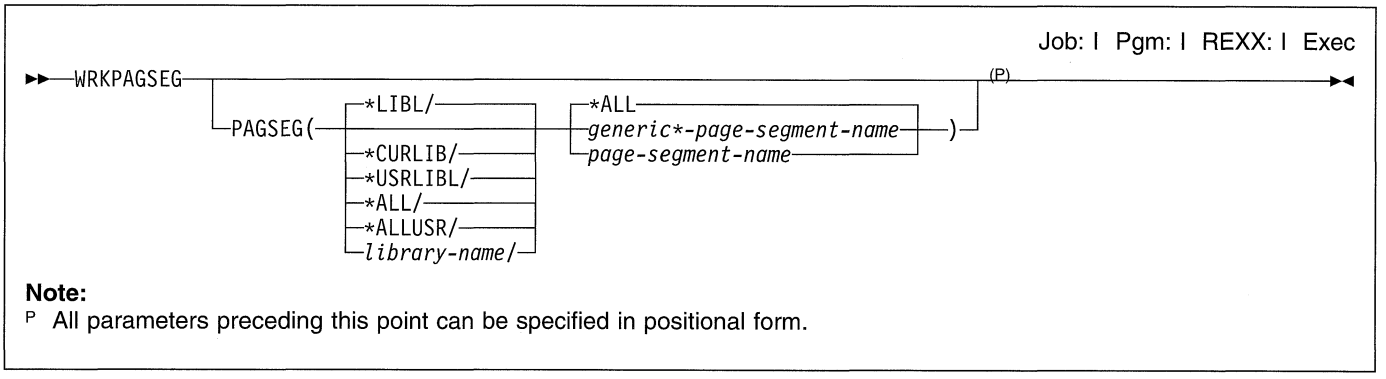
page-definition-name: Specify the name of the page definition. If *LIBL or *USRLIBL is specified as the library name, only the first page definition found with the specified name is shown.

Example

```
WRKPAGDFN  PAGDFN(*CURLIB/P1DFLT)
```

This command searches the current library for the page definition P1DFLT. If P1DFLT does not exist, the Work with Page Definitions display shows a message indicating that an object matching the specified name cannot be found.

WRKPAGSEG (Work with Page Segments) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Page Segments (WRKPAGSEG) command allows you to work with all of the page segments from the system or user libraries (or both).

Optional Parameters

PAGSEG

Specifies the qualified name of the page segment.

The name of the page segment can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All page segments in the libraries identified in the library qualifier are listed.

generic-page-segment-name:* Specify the generic name of the page segment. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

page-segment-name: Specify the name of the page segment that is listed. If *LIBL or *USRLIBL is specified as the library name, only the first page segment found with the specified name is listed.

Examples

Example 1: Searching for a Page Segment

```
WRKPAGSEG  PAGSEG(MYLIB/PAGSEG1)
```

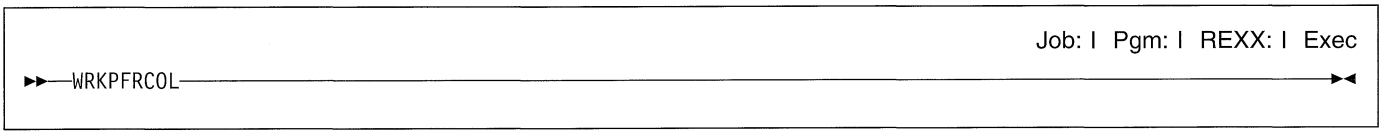
This command searches library MYLIB for a page segment with the name PAGSEG1. If PAGSEG1 is found, the information for that page segment is shown. If a PAGSEG1 does not exist in MYLIB, a message is shown on the WRKPAGSEG display indicating that an object to match the specified name was not found.

Example 2: Searching for a Page Segment

```
WRKPAGSEG  PAGSEG(*LIBL/PAGSEG1)
```

This command searches the library list for the page segment PAGSEG1. Only the first occurrence of PAGSEG1 is listed.

WRKPFRCOL (Work with Performance Collection) Command



Purpose

The Work with Performance Collection (WRKPFRCOL) command shows the user the Work with Performance Collection menu on which automatic performance data collections can be scheduled.

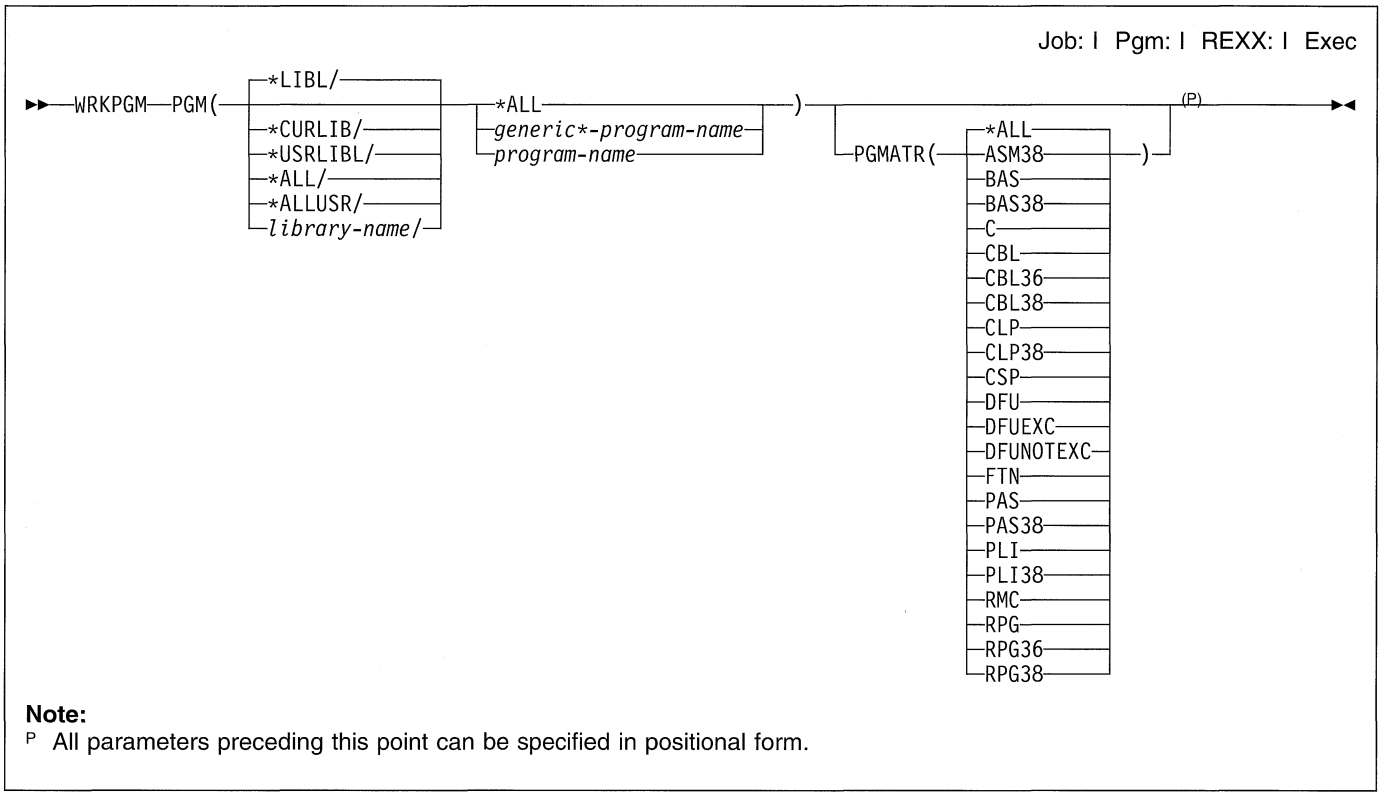
There are no parameters for this command.

Example

```
WRKPFRCOL
```

This command shows the Work with Performance Collection menu on which automatic performance data collections can be added, changed, held, removed, displayed, or released.

WRKPGM (Work with Programs) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Programs (WRKPGM) command allows you to display and work with a list of programs from one or more libraries.

Restrictions: (1) Only the libraries to which you have USE authority will be searched. (2) Only the programs to which you have some authority will be shown on the display. (3) To perform operations on the programs, you must have USE authority to the command used by the operation, and the appropriate authority to the programs on which the operation is to be performed.

Required Parameters

PGM

Specifies that a list of programs in the libraries is shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the programs.

| The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All programs in the libraries identified in the library qualifier are shown (except those libraries for which the user does not have authority).

generic-program-name:* Specify the generic name of
| the program. A generic name is a character string of

one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

program-name: Specify the name of the program shown.

Optional Parameters

PGMATR

Specifies the program attribute of a particular program.

***ALL:** The attributes of all programs are shown.

ASM38: Programs with the attribute ASM38 (assembler programs on the System/38) are shown. These programs use System/38 naming conventions. Programs containing this attribute will not run on the AS/400 system.

BAS: Programs with the attribute BAS (BASIC programs) are shown.

BAS38: Programs with the attribute BAS38 (System/38 BASIC programs used in the System/38 environment) are shown.

C: Programs with the attribute C (C programs) are shown.

CBL: Programs with the attribute CBL (COBOL/400 programs) are shown.

CBL36: Programs with the attribute CBL36 (System/36 COBOL programs used in the System/36 environment) are shown.

CBL38: Programs with the attribute CBL38 (for System/38 COBOL programs) are shown. These programs use System/38 naming conventions.

CLP: Programs with the attribute CLP (control language programs) are shown.

CLP38: Programs with the attribute CLP38 (System/38 COBOL control language programs) are shown. These programs use System/38 naming conventions.

CSP: Programs with the attribute CSP (the root of a CSP/AE application) are shown.

DFU: Programs with the attribute DFU (programs created by the AS/400 data file utility) are shown. These programs use System/38 naming conventions.

DFUEXC: Programs with the attribute DFUEXC (programs created by the System/38 data file utility which can be run using System/38 DFU) are shown. These programs use System/38 naming conventions.

DFUNOTEXC: Programs with the attribute DFUNOTEXC (programs created by the System/38 data file utility (DFU)) are shown. These programs cannot be run using System/38 DFU.

FTN: Programs with the attribute FTN (FORTRAN programs) are shown.

PAS: Programs with the attribute PAS (PASCAL programs) are shown.

PAS38: Programs with the attribute PAS38 (System/38 PASCAL programs) are shown. These programs use System/38 naming conventions.

PLI: Programs with the attribute PLI (programming language one (PL/I) programs) are shown.

PLI38: Programs with the attribute PLI38 (System/38 programming language one (PL/I) programs) are shown. These programs use System/38 naming conventions.

RMC: Programs with the attribute RMC (RM/COBOL-85** for the AS/400) are shown.

RPG: Programs with the attribute RPG (RPG/400* programs) are shown.

RPG36: Programs with the attribute RPG36 (RPG II programs in the System/36 environment) are shown.

RPG38: Programs with the attribute RPG38 (RPG III programs) are shown. These programs use System/38 naming conventions.

Examples

Example 1: Listing Control Language Programs

```
WRKPGM  PGM(MYLIB/*ALL)  PGMATR(CLP)
```

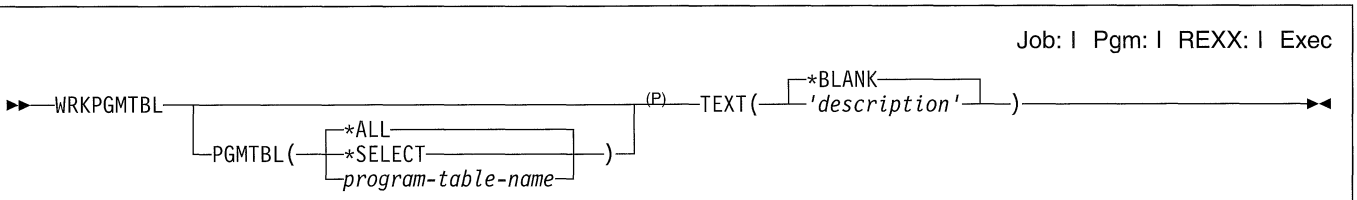
This command lists all the CL programs to which the user has some authority that are stored in library MYLIB.

Example 2: Listing Programs

```
WRKPGM  PGM(AB*)
```

This command lists all of the programs in the library list that have names beginning with AB.

WRKPGMTBL (Work with Program Tables) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Program Tables (WRKPGMTBL) command allows you to create finance program tables. Once they are created, you can add or delete program names in these tables. Several finance program tables can be defined, but each table must have a unique name.

Finance program table updates can be accessed by any finance job that is submitted after all changes are completed.

Restriction: This command is shipped with public *EXCLUDE authority.

Optional Parameters

PGMTBL

Specifies the name of a table that contains finance program IDs and user-associated program names.

***ALL:** The list of existing program tables is shown. From this display, you can create, delete, change, or display program tables.

***SELECT:** The list of existing program tables is shown. This value, which performs the same function as *ALL, is included for compatibility with previous releases.

program-table-name: Specify the name of the program table with which to work.

TEXT

Unused parameter provided for compatibility with previous releases.

Examples

Example 1: Working With All Finance Program Tables

```
WRKPGMTBL  PGMTBL(*SELECT)
```

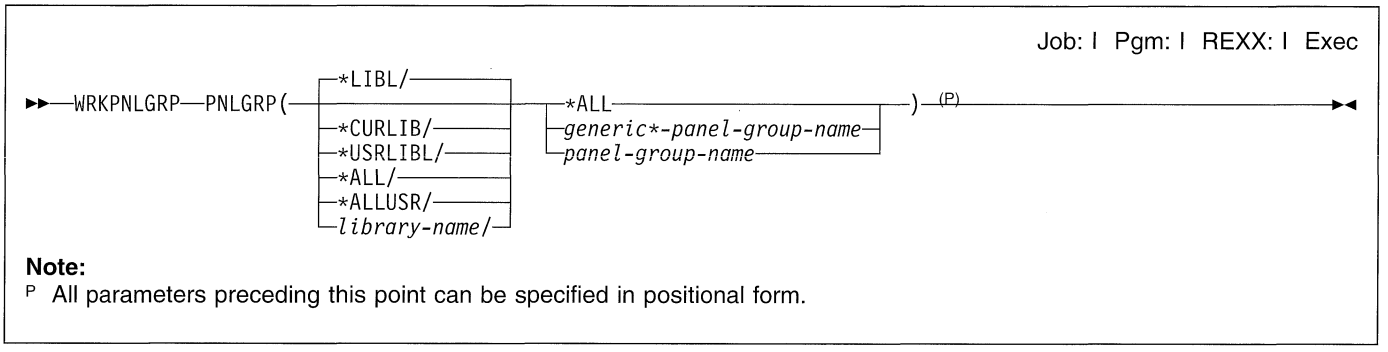
This command allows you to work with all finance program tables. You can create a new table, or select an existing table to change, delete, or display.

Example 2: Working With a Program Table

```
WRKPGMTBL  PGMTBL(PGMTBL1)
```

This command allows you to work with program table PGMTBL1. With this command you can create, change, delete, or display a table.

WRKPNLGRP (Work with Panel Groups) Command



Note:

| (P) All parameters preceding this point can be specified in positional form.

Purpose

The Work with Panel Groups (WRKPNLGRP) command allows you to display and work with a list of panel groups from one or more libraries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the panel groups to which you have some authority will be shown on the display.
3. To perform operations on the panel groups, you must have USE authority to the command used by the operation, and the appropriate authority to the panel groups on which the operation is to be performed.

Required Parameters

PNLGRP

Specifies a list of panel groups in the libraries shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the panel groups.

The name of the panel group can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All panel groups in the libraries identified in the library qualifier are shown.

generic-panel-group-name:* Specify the generic name of the panel group. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

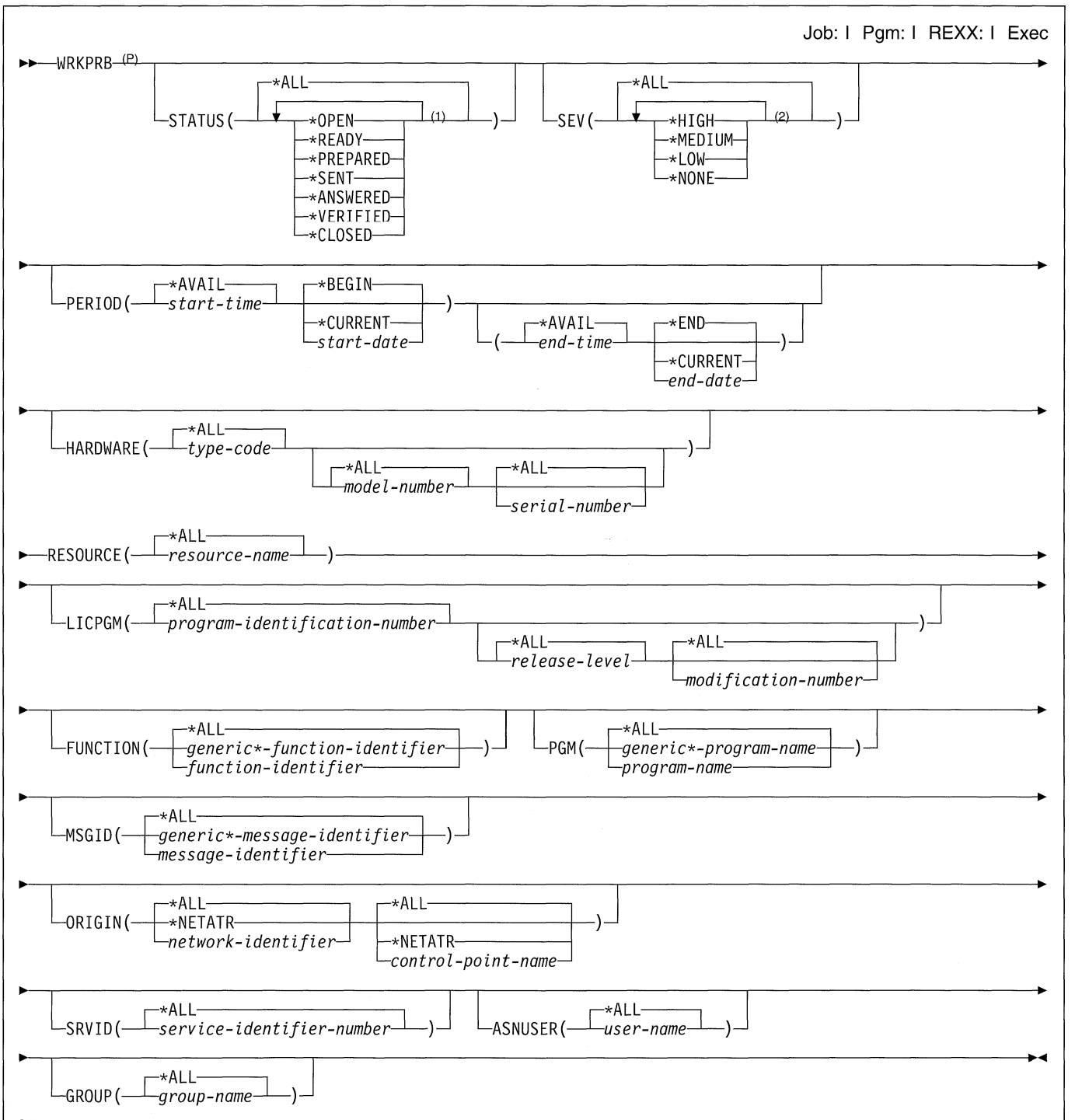
panel-group-name: Specify the name of the panel group that is shown.

Example

```
WRKPNLGRP  PNLGRP(LIB01/ABC*)
```

This command allows you to display and work with a list of panel groups beginning with ABC stored in library LIB01.

WRKPRB (Work with Problems) Command



Notes:

P All parameters preceding this point can be specified in positional form.

1 A maximum of 6 repetitions

2 A maximum of 3 repetitions

Purpose

The Work with Problems (WRKPRB) command allows the user to work with the Work with Problems display. This command is used to work with problems that were detected by the system or detected by the user.

Problems can be deleted from the log by using the Delete Problem (DLTPRB) command.

Restriction: This command is shipped with public *EXCLUDE authority and the QPGMR, QSYSOPR, QSRV, and QSRVBAS user profiles have private authorities to use the command.

Optional Parameters

STATUS

Specifies the status of the problem log entries. The seven status types are as follows:

*OPENED The problem is in OPENED status. The problem has been identified and a problem record has been created.

*READY The problem is in READY status. Problem analysis information has been added to the problem record.

*PREPARED The problem is in PREPARED status. The problem has been prepared for reporting.

*SENT The problem is in SENT status. The problem has been sent to a service provider, but no answer has been returned.

*ANSWERED The problem is in ANSWERED status. An answer was returned by the service provider or added by an operator on this system.

*VERIFIED The problem is in VERIFIED status. The problem has been resolved and the system operator has verified that the problem has been corrected.

*CLOSED The problem is closed.

***ALL:** Problem log entries with any status are shown.

status-type: Specify up to six of the seven status types.

SEV

Specifies the severity level of the problem log entries being shown on the display. The severity level is assigned by the user when the problem is prepared for reporting. The four severity levels are:

1 High
2 Medium
3 Low
4 None

***ALL:** Problem log entries with any severity level are shown.

severity-level: Specify up to three of the four severity levels.

PERIOD

Specifies the time when the problem entry was placed in the log. The period of time is specified by the starting time and date and the ending time and date.

Element 1: Starting Time

***AVAIL:** The logged data that is available for the specified starting date is displayed.

start-time: Specify the time at which the data is logged. The time is specified in 24-hour format with or without a time separator as follows:

- With a time separator, specify a string of 5 or 8 digits where the time separator separates the hours, minutes, and seconds. If this command is entered from the command line, the string must be enclosed in apostrophes. If a time separator other than the separator specified for your job is used, this command fails.
- Without a time separator, specify a string of 4 or 6 digits (hhmm or hhmmss) where **hh** = hours, **mm** = minutes, and **ss** = seconds. Valid values for **hh** range from 00 through 23. Valid values for **mm** and **ss** range from 00 through 59.

Element 2: Starting Date

***BEGIN:** The logged data from the beginning of the log is displayed.

Note: If PERIOD(*BEGIN) is specified, then any time value other than *AVAIL for start-time is ignored.

***CURRENT:** The logged data for the current day, and the time that has elapsed between the specified starting time and ending time for the day, is displayed.

start-date: Specify the date shown. The date must be entered in the format specified by the system values QDATFMT and, if separators are used, QDATSEP.

Element 3: Ending Time

***AVAIL:** The logged data that is available for the specified ending date is displayed.

end-time: Specify the time at which logging of the data ends. See the description of *start-time* for details about how time can be specified.

Element 4: Ending Date

***END:** The last day on which data was logged is shown. If PERIOD(*END) is specified, a time value other than *AVAIL for end-time is ignored.

***CURRENT:** The logged data for the current day, and the time that has elapsed between the specified starting time and ending time for the day, is displayed.

end-date: Specify the last date on which the data is logged. The date must be entered in the format speci-

fied by the system values QDATFMT and, if separators are used, QDATSEP.

HARDWARE

Specifies that only problem log entries identifying the failing device are shown.

Note: If HARDWARE(*ALL) is specified, *ALL must be specified for machine type, model, and serial number.

Element 1: Machine Type

***ALL:** Entries are shown regardless of which device, if any, is identified as failing.

type-code: Specify the 4-character type code of the device.

Element 2: Model Number

***ALL:** All entries identifying the specified type of failing device are shown.

model-number: Specify the 3-character model number of the device.

Element 3: Serial Number

***ALL:** All entries identifying the specified type and model of the failing device are shown.

serial-number: Specify the serial number of the device in one of the following formats where n is a decimal digit ranging from 0 through 9.

- nnnnn
- nnnnnnn
- nn-nnnnn
- nn-nnnnnnn

RESOURCE

Specifies that only problem log entries identifying the failing resource name are shown.

***ALL:** Entries are shown regardless of which resource name, if any, is identified by the problem.

resource-name: Specify the resource name.

LICPGM

Specifies that only problem log entries identifying the failing licensed program are shown.

Note: If LICPGM(*ALL) is specified, then *ALL must be specified for licensed program, release level, and modification number.

Element 1: Licensed Program

***ALL:** All licensed programs are shown regardless of whether any are identified as failing.

program-identification-number: Specify the identification number of the licensed program.

Element 2: Release Level of the Licensed Program

***ALL:** All entries identifying failing licensed programs are shown.

release-level: Specify the release level of the licensed program.

Element 3: Modification Number of the Release

***ALL:** All entries identifying a failing licensed program of the specified licensed program and release are shown.

modification-number: Specify the modification number of the release.

FUNCTION

Specifies that only problem log entries identifying the function identifier are shown. This identifier is present only in user detected problem log entries.

***ALL:** Entries are shown regardless of which function identifier, if any, is identified.

generic-function-identifier:* Specify the generic name of the function identifier. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

function-identifier: Specify the complete function identifier. If blank characters are included, the character string must be enclosed in apostrophes.

PGM

Specifies that only problem log entries identifying the failing program are shown. For machine-detected problems, the failing program is identified by the possible cause with the highest probability of failure.

***ALL:** Entries are shown regardless of which program, if any, is identified.

generic-program-name:* Specify the generic name of the program.

program-name: Specify the name of the program.

MSGID

Specifies that only problem log entries identifying the message identifier are shown. This is the problem message identifier shown in the problem details display. For user detected problems, the message identifier is entered by the user.

***ALL:** Entries are shown regardless of which message identifier is associated with a problem.

generic-message-identifier:* Specify the generic name of the message identifier.

message-identifier: Specify the message identifier.

ORIGIN

Specifies that only problem log entries originating at the specified nodes are shown.

Element 1: Network Identifier

***ALL:** Entries are shown regardless of the network identifier of the origin system of each entry.

***NETATR:** Only entries that originated on systems with the same local network identifier as the one defined in the network attributes for this system are shown.

network-identifier: Specify a network identifier. Only entries originating on systems with this local network identifier are shown.

Element 2: Control Point Name

***ALL:** All entries originating on systems using the network identifier are shown.

***NETATR:** Only entries originating on systems with the same control point name as the one defined in the network attributes are shown.

control-point-name: Specify the name of the control point. Only entries originating on systems with this control point name are shown.

SRVID

Specifies that only problem log entries using the service-assigned number are shown. This number is assigned when the problem is reported to IBM service support.

***ALL:** Entries are shown regardless of which service number, if any, is assigned to the problem.

service-identifier-number: Specify the service identifier number.

ASNUSER

Specifies which problem log entries are displayed.

***ALL:** All program log entries are displayed, regardless of the user assigned to them.

user-name: Specify the user name assigned to the problem log entries to be displayed.

GROUP

Specifies the group in the filter to which the problem is assigned.

***ALL:** All problem log entries are displayed, regardless of the group assigned to them.

group-name: Specify the 10-character problem filter group assigned to the entry.

Note: The values are blank if problem log filtering is not used.

Examples

Example 1: Displaying Entries with Status of OPENED or READY

```
WRKPRB  STATUS(*OPENED *READY) HDW(9347)
```

This command shows the Work with Problems display listing of only those problem entries with a status of OPENED or READY which identify a failing device with type 9347.

Example 2: Displaying Current Day Problem Entries

```
WRKPRB  PERIOD((*AVAIL *CURRENT))
```

This command shows the Work with Problems display listing all problem entries that are created in the log on the current day.

Example 3: Displaying List of Hardware Problems

```
WRKPRB  SEV(1 2)  HARDWARE(9347 001 10-7523489)
```

This command shows a list containing problems with the hardware specified by the user. The user has specified that the command track medium to high levels of severity.

Example 4: Displaying Problems That Have Been Opened

```
WRKPRB  STATUS(*OPENED)  PERIOD((*AVAIL  *CURRENT)
  (120000  *CURRENT))  LICPGM(5728SS1  03  00)
  PGM(QNOPGM)
```

This command shows a list containing problems that have been opened during the period starting at midnight and ending at noon on the current day, and have not yet been analyzed. This command also identifies the specified licensed program identifier and program name as the probable cause of the failure.

Example 5: Displaying a List of Machine-Detected Problems

```
WRKPRB  RESOURCE(TAP01)  MSGID(CPF6788)
```

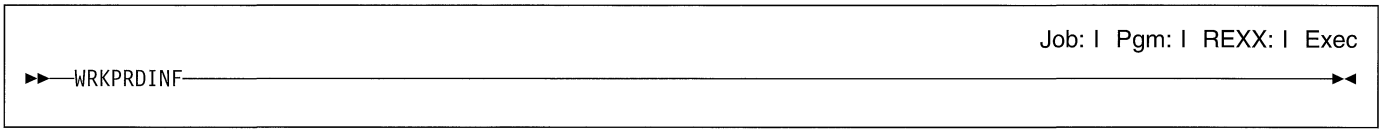
This command shows a list containing machine detected problems that were opened due to the message, CPF6788, having been sent to the system operator message queue and for which a problem analysis was done. The problem analysis was done to determine the resource name of the device suspected of failure, which in this case is device, TAP01. The list of problems includes user-detected problems. To get the user-detected problems, the user specified the resource name and message identifier by using the Analyze Problem (ANZPRB) command.

Example 6: Displaying a List of Reported Problems

```
WRKPRB  SRVID(12345)
```

This command shows a list containing problems that have been reported to an IBM service identifier.

WRKPRDINF (Work with Product Information) Command



Purpose

The Work with Product Information (WRKPRDINF) command allows you to access information available through marketing support systems. When you use this command, a display without prompts is shown, informing you that the AS/400 system is going to start 3270 emulation. If you experience a problem, you can call the Help Desk number shown. If you press the Enter key to continue, a communications session is established.

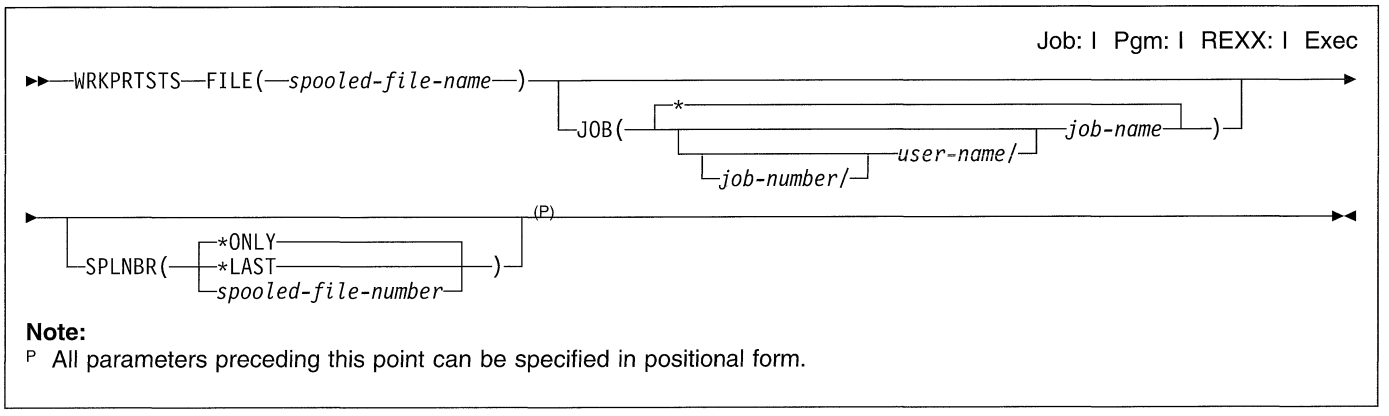
There are no parameters for this command.

Example

```
WRKPRDINF
```

This command allows you to sign on the IBM Information Network.

WRKPRTSTS (Work with Printing Status) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Printing Status (WRKPRTSTS) command allows you to list the status of printing jobs for a specified spooled file.

Required Parameters

FILE

Specifies the name of a file created by a user program and the name of the device file used to create the file.

Optional Parameters

JOB

Specifies the name of the job that created the spooled file. This parameter is valid only if a spooled file name is specified in the FILE parameter. If no job qualifier is given, all of the jobs currently in the system are searched for the simple job name.

A job identifier is a special value or a qualified name with up to three elements. For example:

```
*
job-name
user-name/job-name
job-number/user-name/job-name
```

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** The job from which this WRKPRTSTS command was entered is the job that created the spooled file.

job-name: Specify the name of the job that created the spooled file.

user-name: Specify the name of the user of the job that created the spooled file.

job-number: Specify the number of the job that created the spooled file.

SPLNBR

Specifies the number of the spooled file being processed. This parameter is valid only if a spooled file name is specified in the FILE parameter. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ONLY:** One spooled file from the job has the specified file name. The number of the spooled file is not necessary. If *ONLY is specified and more than one spooled file has the specified file name, a message is sent.

***LAST:** The spooled file with the highest number and the specified file name is used.

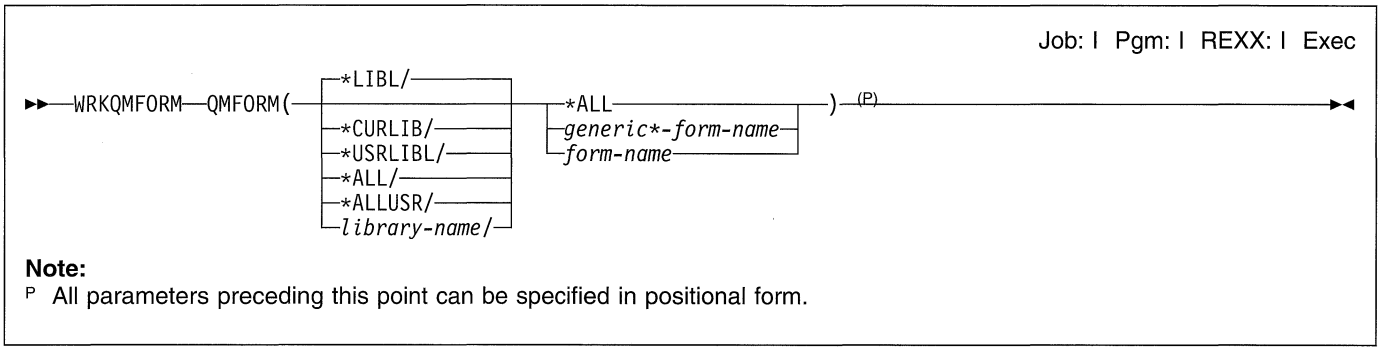
spooled-file-number: Specify the number of the job's spooled file that is on the specified output queue and that is being processed first.

Example

```
WRKPRTSTS  FILE(MYFILE)
```

This command assumes that the user has a spooled file on an output queue that does not have a printer attached to it. When the user specifies this command, the Work with Printing Status display will be shown. One status message that would apply to the file named MYFILE would be, "This file is not associated with a started printer." Depending on the status of MYFILE, other status types could be shown.

WRKQMFORM (Work with Query Management Form) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Query Management Form (WRKQMFORM) command shows a list of query management forms from a user-specified subset of query management form names. From this list, several query management functions are available for working with query management forms.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the query management forms to which you have some authority will be shown on the display.
3. To perform operations on the query management forms, you must have USE authority to the command used by the operation, and the appropriate authority to the query management forms on which the operation is to be performed.

Required Parameters

QMFORM

Specifies the qualified name of the query management form to be shown on the Work with Query Management Form display. A specific query management form or a generic query management form can be specified. Either type can be optionally qualified by a library name.

| The name of the query management form can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All query management forms in the specified libraries are listed on the Work with Query Management Form display.

generic-form-name:* Specify the generic name of the form. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all forms with names that begin with the generic name, and for which the user has authority, are shown. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete form name.

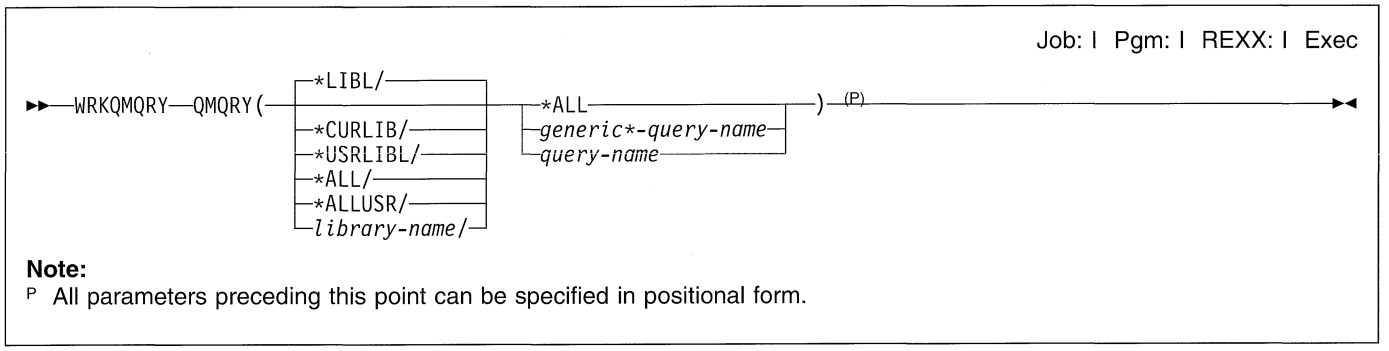
form-name: Specify the name of the query management form to be listed.

Example

```
WRKQMFORM  QMFORM(QGPL/DSP*)
```

This command shows a list of all query management forms in library QGPL that start with DSP.

WRKQMQRY (Work with Query Management Query) Command



Note:

| (P) All parameters preceding this point can be specified in positional form.

Purpose

The Work with Query Management Query (WRKQMQRY) command shows a list of query management queries from a user specified subset of query management query names. From this list, several query management functions are available for working with the query management queries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the query management queries to which you have some authority will be shown on the display.
3. To perform operations on the query management queries, you must have USE authority to the command used by the operation, and the appropriate authority to the query management queries on which the operation is to be performed.

Required Parameters

QMQRY

Specifies the qualified name of the query management query to be shown on the Work with Query Management Queries display. A specific query management query or a generic query management query can be specified. Either type can be optionally qualified by a library name.

| The name of the query management query can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All query management queries in the specified libraries are listed on the Work with Query Management Query display.

generic-query-name:* Specify the generic name of the query. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all queries with names that begin with the generic name, and for which the user has authority, are shown. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete query name.

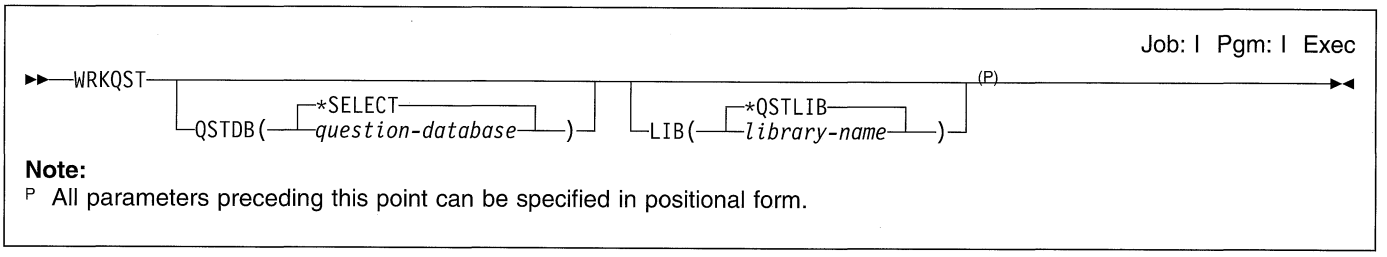
query-name: Specify the name of the query management query to be listed.

Example

```
WRKQMQRY  QMQRY(QGPL/DSP*)
```

This command shows a list of all query management queries in library QGPL that start with DSP.

WRKQST (Work with Questions) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Question (WRKQST) command allows a user to review the questions asked. More information is in the *Q & A Database Coordinator's Guide*.

Restriction: The user must have read authority to the database.

Optional Parameters

QSTDB

Specifies the Question-and-Answer (Q & A) database with which to work.

***SELECT:** The user is asked to specify a Q & A database. If only one Q & A database exists on the system, it is the default.

question-database: Specify the name of the Q & A database with which to work.

LIB

Specifies the name of the library that contains the Q & A database.

***QSTLIB:** The library containing the specified Q & A database is searched. If *SELECT is specified on the QSTDB parameter, any Q & A database in any library for which the user is authorized can be selected.

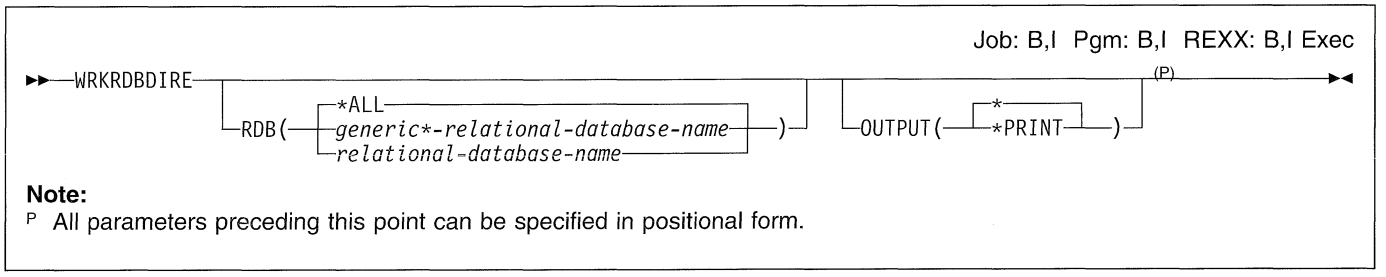
library-name: Specify the name of the library to be searched. If *SELECT is specified on the QSTDB parameter, any database in the library for which the user is authorized can be selected.

Example

```
WRKQST
```

This command shows the Work with Questions You Asked display. If more than one database is available for selection, the Select Q and A Database display is shown first.

WRKRDBDIRE (Work with Relational Database Directory Entry) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Relational Database Directory Entries (WRKRDBDIRE) command allows you to show and work with one or more entries from the relational database directory.

With this command, you can do the following with the relational database directory:

- Add new entries
- Change existing entries
- Remove entries
- Show the details of an entry
- Print an entry
- Print a list of all entries

Optional Parameters

RDB

Specifies the name of the relational database entry to be shown and worked with.

***ALL:** All entries in the relational database directory are shown.

generic-relational-database-name:* Specify the generic name of the relational database entry. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

relational-database-name: Specify a maximum of 18 characters for the name of the relational database entry.

OUTPUT

Specifies whether the relational database information is shown at the requesting work station or is printed.

More information on this parameter is in "Appendix A. Expanded Parameter Descriptions" in the *CL Reference*.

***:** Output requested by an interactive job is shown. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The requested data is printed with the job's spooled output.

Examples

Example 1: Displaying All Directory Entries

```
WRKRDBDIRE
```

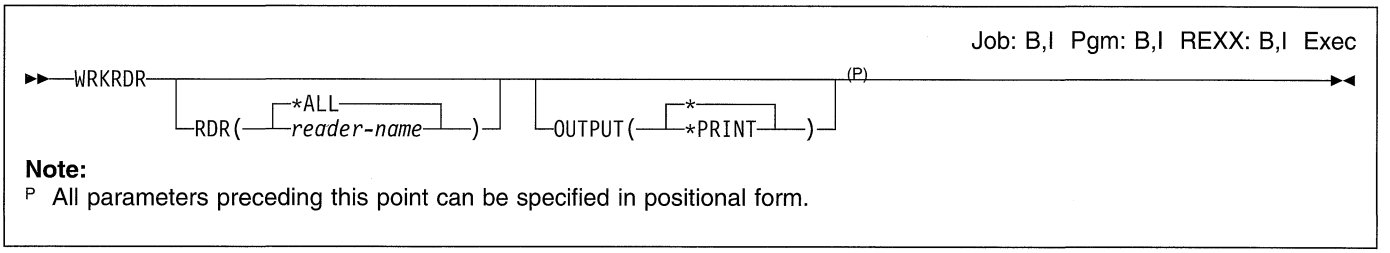
This command shows a list of all relational database directory entries and allows you to work with them.

Example 2: Directing Information to a Database File

```
WRKRDBDIRE   RDB(YOURRDB)  OUTPUT(*PRINT)
```

This command directs the information from the relational database directory entry YOURRDB to a printer file.

WRKRDR (Work with Readers) Command



Purpose

The Work with Readers (WRKRDR) command allows you to work with the overall status of all readers or the detailed status of a specific reader. The status of the readers may change while the command is in process.

Optional Parameters

RDR

Specifies whether overall status is given for all readers or detailed status is given for a specified reader.

***ALL:** The overall status of all readers is displayed.

reader-name: Specify the name of the reader whose detailed information is displayed.

OUTPUT

Specifies whether the output from the command is displayed at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

Examples

Example 1: Working With All Readers

```
WRKRDR
```

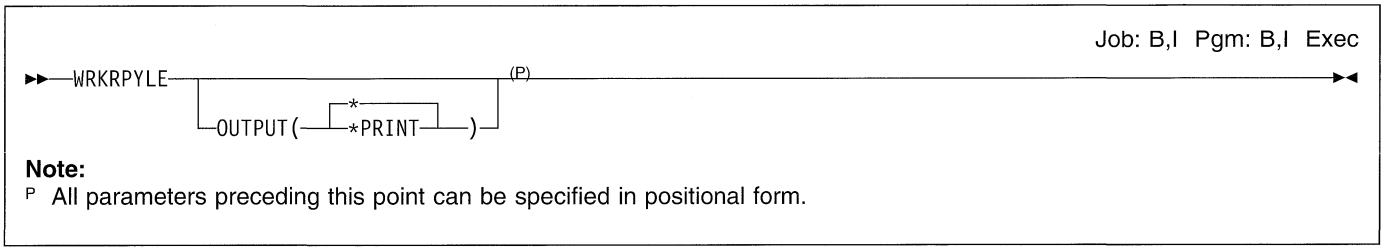
This command allows the user to work with the names of all readers, their types, device files, and status.

Example 2: Working With One Reader

```
WRKRDR  RDR(DISKREAD)
```

This command allows the user to work with the detailed information about the reader DISKREAD.

WRKRPYLE (Work with System Reply List Entries) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with System Reply List Entries (WRKRPYLE) command can be used to display or print all the reply entries currently in the system message reply list. The system reply list contains replies that are automatically sent in response to inquiry messages.

The following information is shown for each reply in the system reply list:

- Sequence number
- Message ID
- Reply
- Dump indication
- Compare value
- Compare start position

From this display the user can add, change, or remove individual reply list entries.

The reply list is only used when an inquiry message is sent by a job that has the inquiry message reply attribute of the system reply list specified (*SYSRPYL is specified on the INQMSGRPY parameter). The INQMSGRPY attribute can be changed with the CHGJOB command.

The user can add reply list entries with the Add Reply List Entry (ADDRPYLE) command. Specific attributes of a reply list entry can be changed with the Change Reply List Entry (CHGRPYLE) command. Each reply list entry remains in the list until it is removed by the Remove Reply List Entry (RMVRPYLE) command.

Optional Parameters

OUTPUT

Specifies whether the reply list entries are printed or displayed. If the job is a batch job or if *PRINT is specified, the system reply list is listed in the output file. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

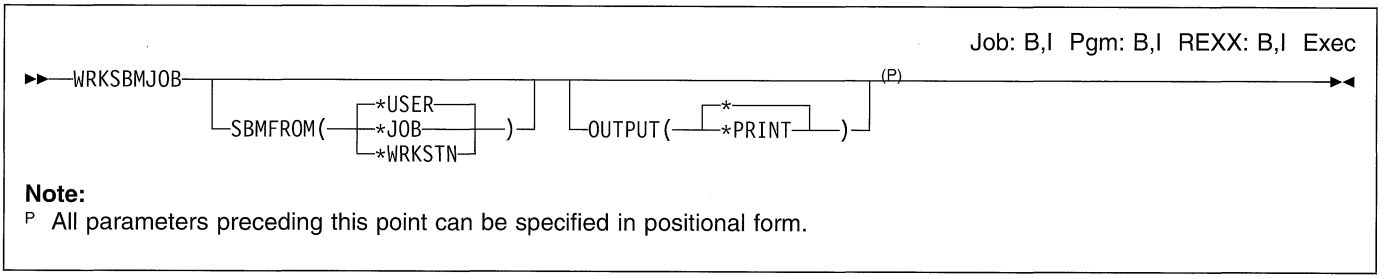
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKRPYLE  OUTPUT(*PRINT)
```

This command allows the user to print the entries in the system reply list.

WRKSBMJOB (Work with Submitted Jobs) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Submitted Jobs (WRKSBMJOB) command allows you to work with the status of all jobs submitted at a work station, in a job, or under a user profile. Jobs submitted with DSPSBMJOB(*NO) specified on the Submit Job (SBMJOB), Submit Database Jobs (SBMDBJOB), or Submit Diskette Jobs (SBMDKTJOB) commands are not displayed by this command.

Optional Parameters

SBMFROM

Specifies the type of submitted jobs that are displayed. Jobs of the specified type that were submitted with the parameter DSPSBMJOB(*NO) specified on the SBMJOB, SBMDBJOB, or SBMDKTJOB commands are not included.

***USER:** Jobs that were submitted from a job having the same user profile as the job in which this command is entered are displayed.

***JOB:** Jobs that were submitted from the same job in which this command is entered are displayed.

***WRKSTN:** Jobs that were submitted from the same work station at which this command is entered are displayed.

OUTPUT

Specifies whether output is displayed at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

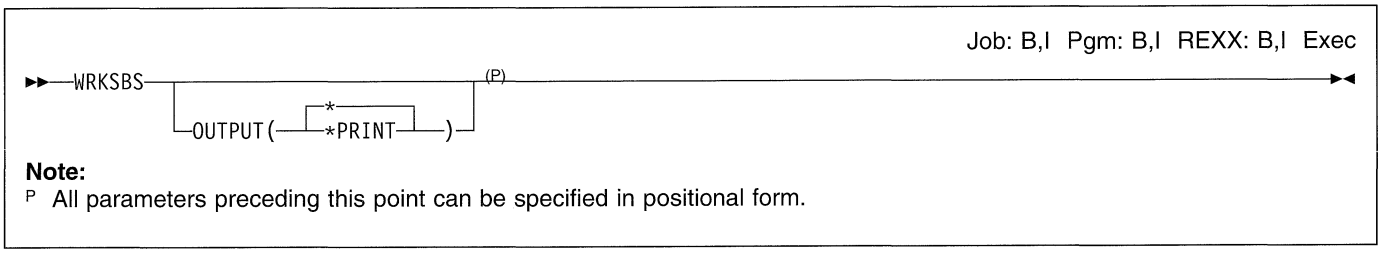
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKSBMJOB  SBMFROM(*USER)
```

This command allows you to work with or change a list of jobs that are submitted by a job running under the same user profile as the job where this command is run.

WRKSBS (Work with Subsystems) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Subsystems (WRKSBS) command allows you to work with each active subsystem in the system. Also, if one of the subsystems shown on the system display is selected, additional information listing all of the jobs active in that subsystem are shown.

Optional Parameters

OUTPUT

Specifies whether output is shown at the requesting work station or printed with the job's spooled output.

More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

*: Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

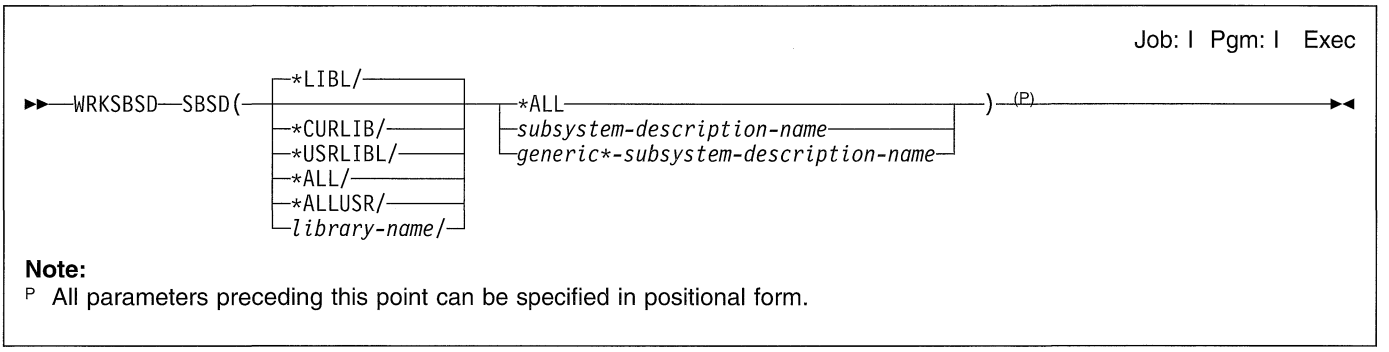
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKSBS
```

This command, entered from a work station, allows you to work with active subsystems. If the command is entered from a batch job, the output information is directed to the job's output spooling queue and printed.

WRKSBSD (Work with Subsystem Descriptions) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Subsystem Descriptions (WRKSBSD) command allows you to work with a list of subsystem descriptions.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the subsystem descriptions to which you have some authority will be shown on the display.
3. To perform operations on the subsystem descriptions, you must have USE authority to the command used by the operation, and the appropriate authority to the subsystem descriptions on which the operation is to be performed.
4. You must have object operational and object management authorities to the subsystem descriptions.

Required Parameters

SBSD

Specifies the qualified name of the subsystem description that is shown.

| The name of the subsystem description can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** Specifies that all subsystem descriptions, which are in the specified libraries are searched.

subsystem-description-name: Specify the name of the subsystem description being shown.

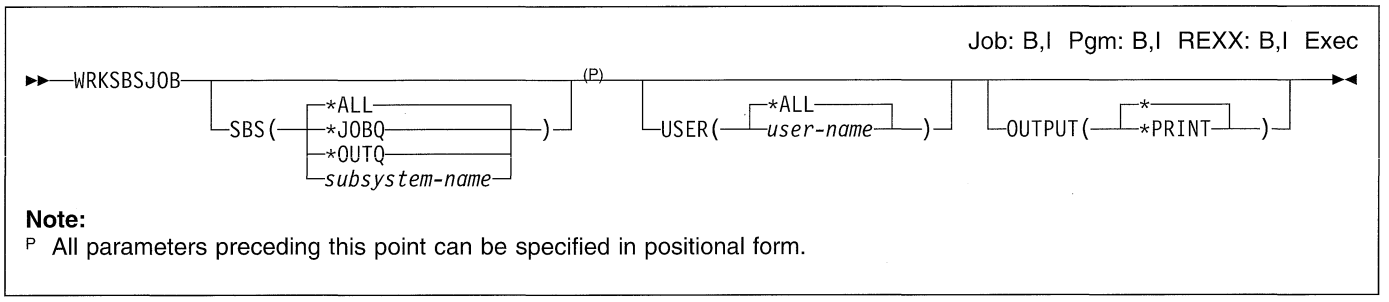
generic-subsystem-description-name:* Specify the
| generic name of the subsystem description. A generic
| name is a character string of one or more characters fol-
| lowed by an asterisk (*); for example, ABC*. The
| asterisk substitutes for any valid characters. A generic
| name specifies all objects with names that begin with the
| generic prefix for which the user has authority. If an
| asterisk is not included with the generic (prefix) name,
| the system assumes it to be the complete object name.
| If the complete object name is specified, and multiple
| libraries are searched, multiple objects can be worked
| with only if *ALL or *ALLUSR library values can be spec-
| ified for the name. For more information on the use of
| generic functions, refer to "Rules for Specifying Names."

Example

```
WRKSBSD  SBSD(LIB6/ORDER*)
```

This command allows you to work with a list of all subsystem descriptions in library LIB6 that start with ORDER. From this list, several SBSD related commands can be performed.

WRKSBSJOB (Work with Subsystem Jobs) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Subsystem Jobs (WRKSBSJOB) command allows the user to work with jobs being processed by subsystems in the system, and the jobs that are on a job or output queue. If a user name is specified, only those jobs belonging to the specified user can be worked with or changed. Also, if one of the jobs shown on the subsystem display is selected, additional information about that job can be displayed.

Optional Parameters

SBS

Specifies the name of the subsystem (or all subsystems) for which the job name and job status of each job currently active in the subsystem, or jobs on a job queue or output queue, are worked with.

Note: This does not include system jobs or subsystem monitor jobs, but it does include readers and writers.

***ALL:** All jobs in all subsystems have their job information displayed. Jobs that are on job queues and on output queues are also worked with.

***JOBQ:** Jobs that are on a job queue are worked with.

***OUTQ:** Jobs that are on an output queue are worked with.

subsystem-name: Specify the name of the subsystem. All active jobs in this subsystem are worked with.

USER

Specifies the name of the user whose jobs are worked with.

***ALL:** All jobs being processed under all user names are worked with.

user-name: Specify a user name. All jobs with this user name are worked with.

OUTPUT

Specifies whether the output is displayed at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

Examples

Example 1: Working With All Jobs

```
WRKSBSJOB
```

This command, entered from a work station, allows the user to work with all jobs in all subsystems, and the jobs on the job queues and output queues.

Example 2: Working With One User's Job

```
WRKSBSJOB  SBS(QBATCH)  USER(JONES)
```

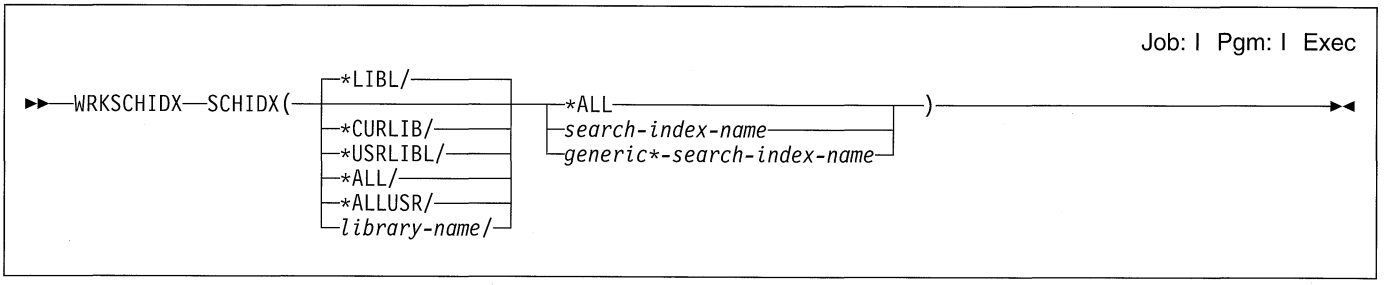
This command allows the user to work with all jobs in the QBATCH subsystem that belong to the user profile of the user named JONES.

Example 3: Working With Jobs on an Output Queue

```
WRKSBSJOB  SBS(*OUTQ)
```

This command allows the user to work with jobs that are on an output queue.

WRKSCHIDX (Work with Search Indexes) Command



Purpose

The Work with Search Indexes (WRKSCHIDX) command allows you to display and work with a list of search indexes from one or more libraries.

Restrictions:

1. Only the libraries to which you have *USE authority are searched.
2. Only the search indexes to which you have some authority are shown on the display.
3. To perform operations on the search indexes, you must have object operational authority.

Required Parameters

SCHIDX

Specifies a list of search indexes in the libraries that are shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the search indexes.

Depending on the library qualifier specified or assumed, the following libraries (for which the user has authority) are searched for the specified search indexes:

The name of the search index can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB     #DFULIB     #RPGLIB     #SEULIB
#COBLIB     #DSULIB     #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX       QPFRDATA    QUSER38
QGPL        QRCL        QUSRSYS
QGPL38      QS36F       QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All search indexes in the libraries identified in the library qualifier are shown.

search-index-name: Specify the name of the search index that is shown.

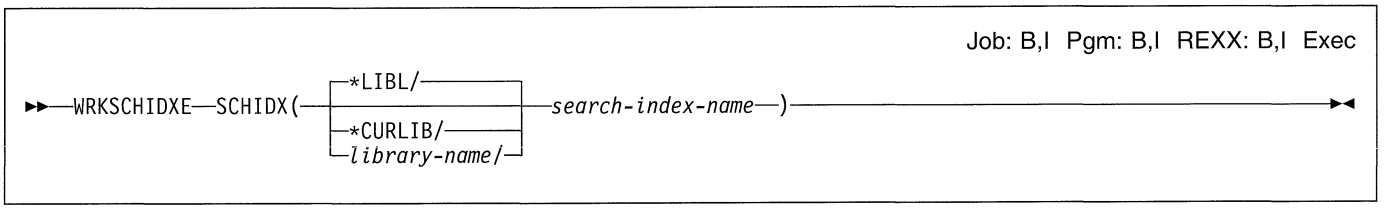
generic-search-index-name:* Specify the generic name of the search index. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Example

```
WRKSCHIDX   SCHIDX(*ALL)
```

This command lists all the search indexes in the library list.

WRKSCHIDXE (Work with Search Index Entries) Command



Purpose

The Work with Search Index Entries (WRKSCHIDXE) command allows you to display and change a search index.

Restriction: You must have authority for the WRKSCHIDXE command and *CHANGE authority for the search index.

Required Parameters

SCHIDX

Specifies the qualified name of the search index being changed.

The name of the search index can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

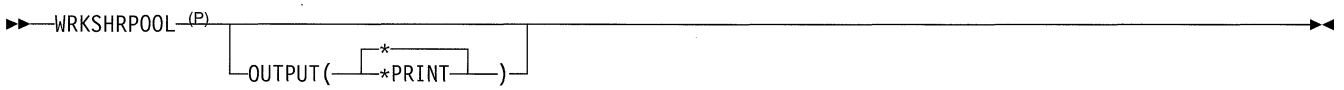
search-index-name: Specify the name of the search index.

Example

```
WRKSCHIDXE   SCHIDX(PAYROLL)
```

This command displays the panel group objects added to the search index PAYROLL. The search index is found by searching the library list (*LIBL default value).

WRKSHRPOOL (Work with Shared Storage Pools) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Shared Storage Pools (WRKSHRPOOL) command allows you to work with shared storage pools including the machine pool and base pool. The pool size and activity level can be changed by typing over the values on the display.

Optional Parameters

OUTPUT

Specifies whether the output from the command is displayed at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

*: Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

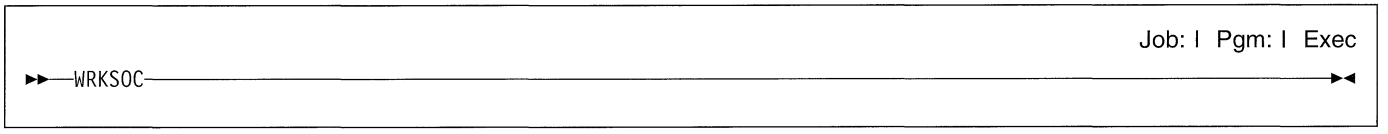
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKSHRPOOL  OUTPUT(*)
```

This command displays shared storage pool information.

WRKSOC (Work with Sphere of Control) Command



Purpose

The Work with Sphere of Control (WRKSOC) command allows you to work with the primary sphere of control by adding or deleting control point names.

More information on sphere of control is in the *Alerts and DSNX Guide*.

Restriction: The user must have use authority for the sphere of control.

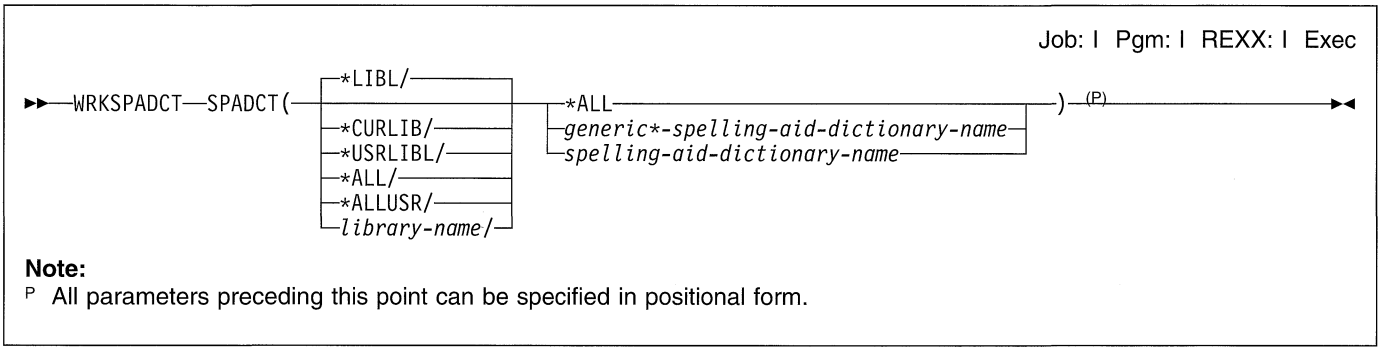
There are no parameters for this command.

Example

```
WRKSOC
```

This command allows you to work with the Work with Sphere of Control display.

WRKSPADCT (Work with Spelling Aid Dictionaries) Command



Note:
| (P) All parameters preceding this point can be specified in positional form.

Purpose

The Work with Spelling Aid Dictionaries (WRKSPADCT) command allows you to display and work with a list of spelling aid dictionaries from one or more libraries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the spelling aid dictionaries to which you have some authority will be shown on the display.
3. To perform operations on the spelling aid dictionaries, you must have USE authority to the command used by the operation, and the appropriate authority to the spelling aid dictionaries on which the operation is to be performed.

Required Parameters

SPADCT

Specifies a list of spelling aid dictionaries in the libraries that are shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the spelling aid dictionaries.

| The name of the spelling aid dictionary can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All spelling aid dictionaries in the libraries identified in the library qualifier are shown.

| *generic*-spelling-aid-dictionary-name:* Specify the generic name of the spelling aid dictionary. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

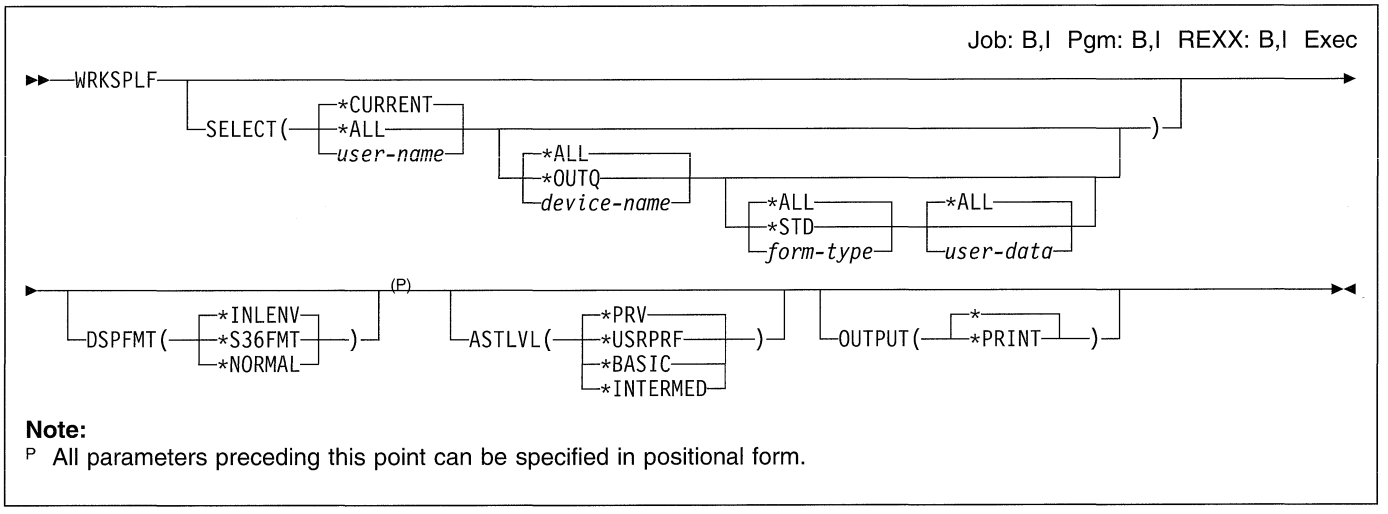
spelling-aid-dictionary-name: Specify the name of the spelling aid dictionary that is shown.

Example

```
WRKSPADCT  SPADCT(*ALL)
```

This command allows you to display the spelling aid dictionaries in the library list.

WRKSPLF (Work with Spooled Files) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Spooled Files (WRKSPLF) command allows the user to work with the spooled files on the system that meet the selection values specified by the SELECT keyword. You can either display or print all the spooled files on the system or a selected subset of them.

Performance Considerations: The amount of time needed to show the list of spooled files is directly proportional to the number of spooled files in the list. On a system with a large number of spooled files, it may be necessary to use the SELECT parameter to view a subset of the system's spooled files.

Optional Parameters

SELECT

Specifies which group of files are selected to be shown or printed. Four positional values can be specified to select the files:

- The user that created the file
- The device for which the file is queued
- The form type specified
- The user data tag associated with the file

Only files that meet each of the requirements are selected.

Element 1: User Values

***CURRENT:** Only files created by the user running this command are selected.

***ALL:** Files created by all users are selected.

user-name: Specify the user who created the files being selected.

Element 2: Device Values

***ALL:** Files on any device-created or user-created output queue are selected.

***OUTQ:** All files on any user-created output queue are selected. A user-created output queue is any output queue that is not automatically created by a device. A user-created output queue does not generally have the same name as a device, but if it does, it does not reside in library QUSRSYS.

device-name: Specify a device name. Only files on the device created output queue for that device are selected. A device created output queue is one that has the same name as a device and resides in the QUSRSYS library. Unless it already exists, it will automatically be created by the system when the device is created. A device created output queue cannot be deleted.

Element 3: Form Type Values

***ALL:** Files for all form types are selected.

***STD:** Only files that specify the standard form type are selected.

form-type: Specify the form type of the queued files being selected.

Element 4: User Data Values

***ALL:** Files with any user data tag specified are selected.

user-data: Specify the user data tag of the queued files being selected.

DSPFMT

Specifies the format and terminology used on the displays that result from running this command.

Note: If the System/36 display format is used, the OUTPUT parameter is ignored and the data is presented at the user's display station.

***INLENV:** The format and terminology that is used is determined by the SPCENV (special environment) value specified in the user profile. If the SPCENV value is *S36, the System/36 terminology is used. All other

values result in the AS/400 system terminology being used.

***S36FMT:** System/36 terminology is used to display the information.

***NORMAL:** AS/400 system terminology is used to display the information.

ASTLVL

Specifies which user interface to use.

***PRV:** The previous user interface is used.

***USRPRF:** The user interface specified in the user profile is used.

***BASIC:** The Operational Assistant user interface is used.

***INTERMED:** The system interface is used.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

Examples

Example 1

```
WRKSPLF  SELECT(*ALL *ALL *ALL *ALL)  DSPFMT(*NORMAL)
```

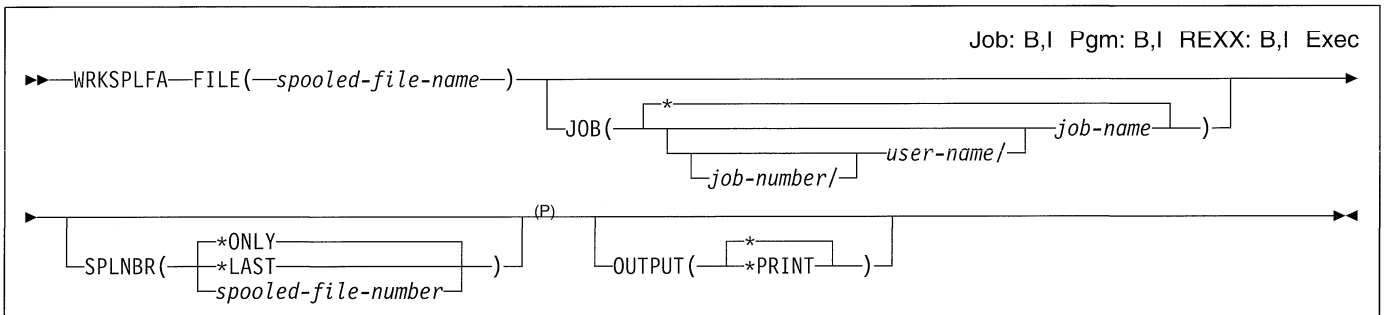
This command allows you to work with all of the spooled files on the system. The AS/400 system terminology used is determined by the value on the SPCENV parameter in the user's profile. The information goes to the display by default.

Example 2

```
WRKSPLF  SELECT(CASMITH *ALL *ALL MEMO)
```

This command allows you to work with all of the spooled files on the system for the user named CASMITH who has MEMO specified in the user data for the spooled file.

WRKSPLFA (Work with Spooled File Attributes) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Spooled File Attributes (WRKSPLFA) command allows you to work with the current attributes of the specified spooled file.

The attributes can be shown after the file is opened and while its file entry is still on the output queue.

Required Parameters

FILE

Specifies the name of the spooled file to have its attributes shown.

JOB

Specifies the qualified name of the job and consists of as many as three elements. For example:

job-name
user-name/job-name
job-number/user-name/job-name

*N may be used in place of the user-name element to maintain position in the sequence. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** The job that entered this WRKSPLFA command is the job that created the spooled file.

job-name: Specify the name of the job that created the spooled file. If no job qualifier is given, all jobs currently in the system are searched for the simple job name. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

Optional Parameters

SPLNBR

Specifies the unique number of the spooled file in the job whose attributes are being shown. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***ONLY:** One spooled file from the job has the specified file name. The number of the spooled file is not necessary. If *ONLY is specified and more than one spooled file has the specified file name, a message is sent.

***LAST:** The spooled file with the highest number and the specified file name is used.

spooled-file-number: Specify the number of the spooled file with the specified file name whose attributes are shown.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** The output is shown if requested by an interactive job or printed if requested by a batch job.

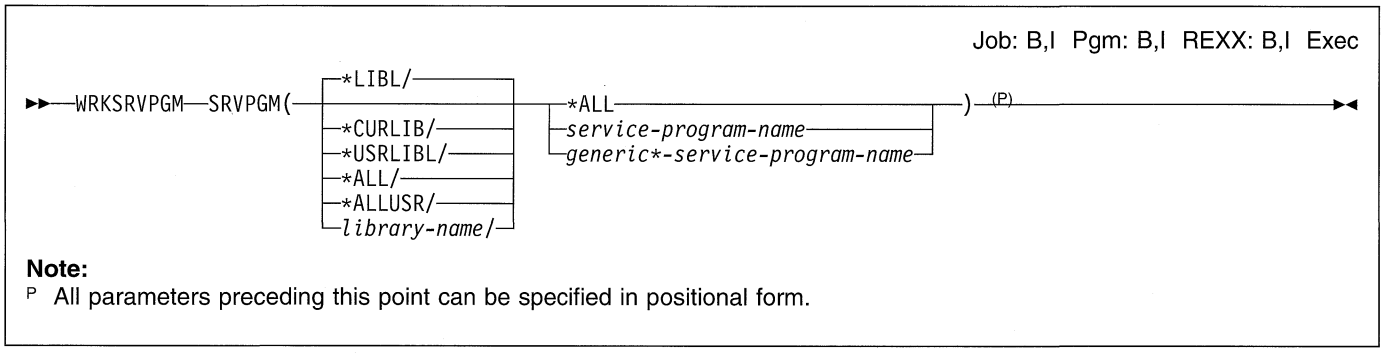
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKSPLFA  FILE(QPRINT)  OUTPUT(*PRINT)
```

This command allows you to work with a file containing the current attributes of the spooled file QPRINT and sends it to the job's output queue to be printed. The job that entered this command must have produced only one output file named QPRINT.

WRKSRVPGM (Work with Service Programs) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Service Program (WRKSRVPGM) command allows you to display and work with a list of service programs from one or more libraries.

Restrictions:

1. Only the libraries to which you have *USE authority are searched.
2. Only the service programs to which you have authority are shown on the display.
3. To perform operations on the service programs, you must have *USE authority to the command used by the operation and the appropriate authority to the service programs on which the operation is to be performed.

Required Parameters

SRVPGM

Specifies the search process for service programs to be placed in the list. All service programs with names corresponding to the specified parameter value, and for which the user has authority, are shown.

The name of the service program list can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All service programs in the libraries identified in the library qualifier are shown.

service-program-name: Specify the name of the service program shown.

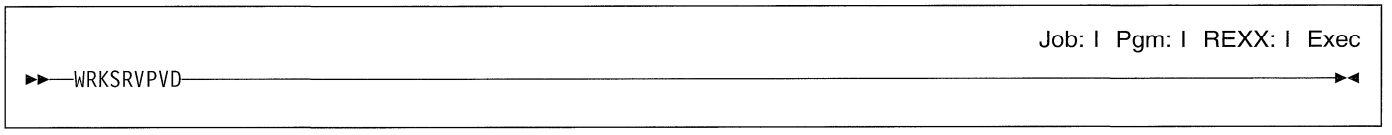
generic-service-program-name:* Specify the generic name of the service program being shown. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. If a generic name is specified, then all service programs with names that begin with the generic name, and for which the user has authority, are shown. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete service program name.

Example

```
WRKSRVPGM   SRVPGM(COMPLETE)
```

This command displays a list of service programs named COMPLETE.

WRKSRVPVD (Work with Service Providers) Command



Purpose

The Work with Service Providers (WRKSRVPVD) command allows you to work with a list of service providers to whom you may report service problems and send PTF orders. From this menu, you can add, change, copy, delete, or display service providers from the list.

Restriction: This command is shipped with public *EXCLUDE authority and the QSRV and QSRVBAS user profiles have private authorities to use the command.

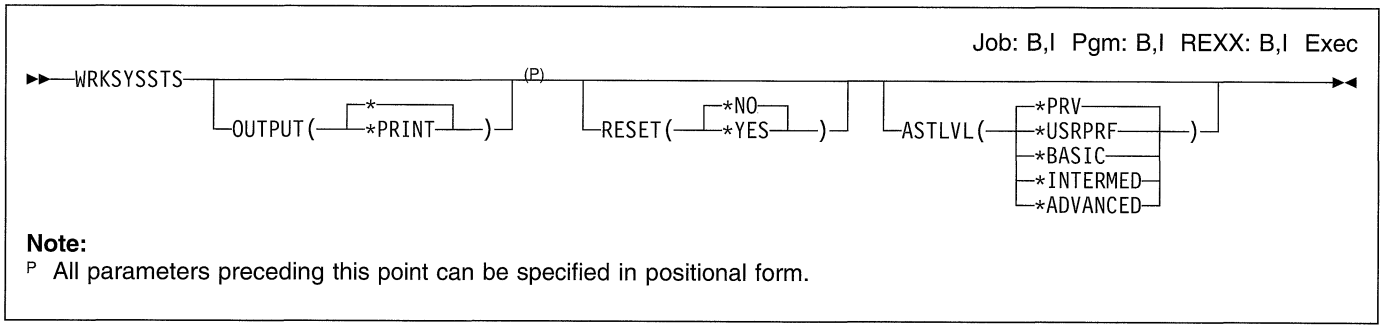
There are no parameters for this command.

Example

```
WRKSRVPVD
```

This command displays the current list of service providers.

WRKSYSSTS (Work with System Status) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with System Status (WRKSYSSTS) command allows you to work with information about the current status of the system. It displays the number of jobs currently in the system, the total capacity of the system auxiliary storage pool (ASP), the percentage of the system ASP currently in use, the amount of temporary storage currently in use, the percentage of machine addresses used, and statistical information related to each storage pool that currently has main storage allocated to it. The statistical information is gathered during an identified time interval (shown as the elapsed time); the data can either be updated by extending the measurement time interval, or it can be restarted to show statistics for the interval starting with the previous display.

Optional Parameters

OUTPUT

Specifies whether the output is displayed at the requesting work station or printed with the job's spooled output. More information on this parameter is in
| Appendix A, "Expanded Parameter Descriptions."

*: Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

RESET

Specifies whether system status statistics fields are reset to zero, as if this is the first occurrence of the WRKSYSSTS command in this job.

***NO:** The system status statistics are not reset.

***YES:** The system status statistics are reset.

ASTLVL

Specifies which user interface to use.

***PRV:** The previously used assistance level is presented.

***USRPRF:** The assistance level defined in the user profile is presented.

***BASIC:** The Operational Assistant user interface is presented.

***INTERMED:** The system user interface is presented.

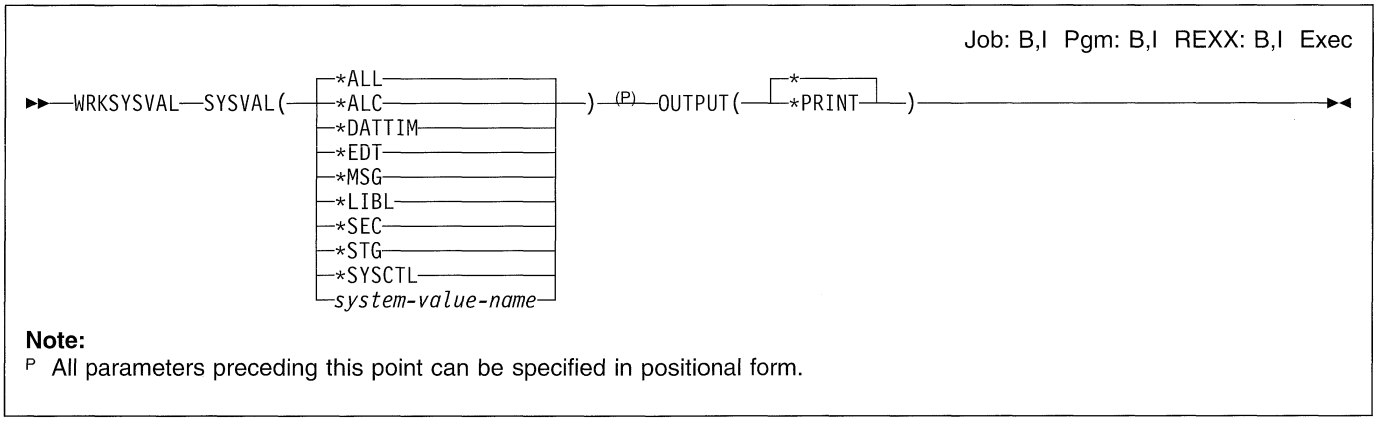
***ADVANCED:** The system user interface expert mode is presented.

Example

```
WRKSYSSTS  OUTPUT(*PRINT)
```

This command allows you to work with the system status information and directs it to the job's output spooling queue for printing. If OUTPUT(*) is specified instead and the command was entered from a work station, the information about the system is displayed at the work station.

WRKSYSVAL (Work with System Values) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with System Values (WRKSYSVAL) command allows the user to display or print a list of system values and change them. System values are provided as part of the system. They are used by the system to control certain operations in the operating system and to communicate the status of certain conditions to the user. Changes to some system values take effect immediately; however, some do not take effect until new jobs are started, and others do not take effect until the next initial program load (IPL). Note that if a change is made to a date or time system value during any operation that measures the length of time, a negative value may be set if the end time is less than the start time. More information about system values is in the *Work Management Guide*.

Restriction: You must have all object (*ALLOBJ) and security administrator (*SECADM) authorities to change security related system values.

Required Parameters

SYSVAL

Specifies the name of the system value whose value is being displayed, printed, or changed. Most of the system values can be specified; however, some values cannot be changed by this command. More information on which values can be specified is in the *Work Management Guide*.

Double-Byte Character Set Considerations:

QIGC is the system value indicating whether the double-byte character set (DBCS) version of the operating system is installed. If the system value for QIGC is 1, then the DBCS version of the operating system is installed. If the system value for QIGC is 0, the DBCS version of the operating system is not installed.

***ALL:** All system values are displayed or printed, depending on what the user specifies on the OUTPUT parameter.

***ALC:** Allocation system values are displayed.

***DATTIM:** Date and time system values are displayed.

***EDT:** Editing system values are displayed.

***MSG:** Message and logging system values are displayed.

***LIBL:** Library list system values are displayed.

***SEC:** Security system values are displayed.

***STG:** Storage system values are displayed.

***SYSCTL:** System control system values are displayed.

system-value-name: Specify the name of the system value with which to work.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** The system value list is displayed if requested by an interactive job or is printed if requested by a batch job.

***PRINT:** The system value list is printed.

Note: The values specified for the SYSVAL and TYPE parameters affect the volume of information that is printed.

Examples

Example 1: Displaying Date and Time System Values

```
WRKSYSVAL  SYSVAL(*DATTIM)
```

This command displays a list of all the date and time system values.

Example 2: Displaying Security System Values

```
WRKSYSVAL  SYSVAL(*SEC)
```

This command displays a list of all the security system values.

Example 3: Displaying a Control Subsystem Description System Value

```
WRKSYSVAL  QCTLSBSD
```

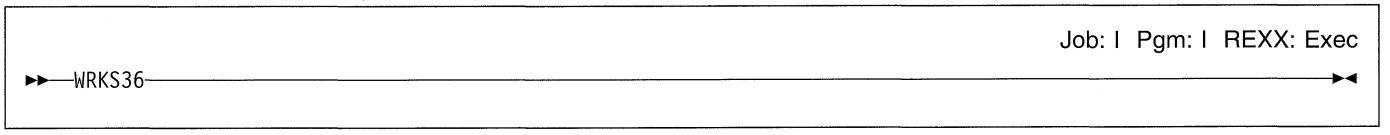
This command displays the control subsystem description (CTLSBSD) system value, QCTLSBSD. Note that the TYPE parameter is left blank.

Example 4: Printing System Values

```
WRKSYSVAL  OUTPUT(*PRINT)
```

This command prints a list of all system values and their current values.

WRKS36 (Work with System/36) Command



Purpose

The Work with System/36 (WRKS36) command shows the Work with System/36 Environment Configuration display.

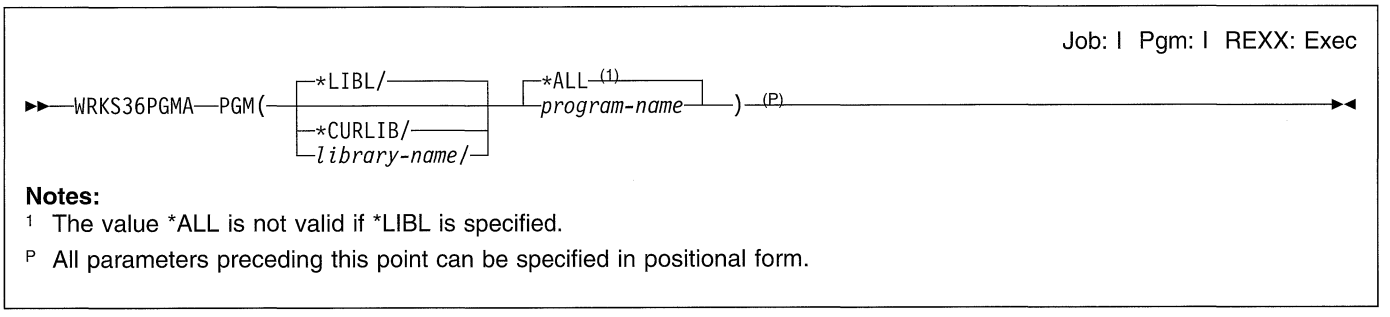
There are no parameters for this command.

Example

```
WRKS36
```

This command shows the Work with System/36 Environment Configuration display.

WRKS36PGMA (Work with System/36 Program Attributes) Command



Notes:

1 The value *ALL is not valid if *LIBL is specified.

P All parameters preceding this point can be specified in positional form.

Purpose

The Work with System/36 Program Attributes (WRKS36PGMA) command shows the Work with System/36 Program Attributes display. The display is a list of programs from a specified library.

Required Parameters

PGM

Specifies the qualified name of the program.

The name of the program can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

***ALL:** All programs in the library are shown.

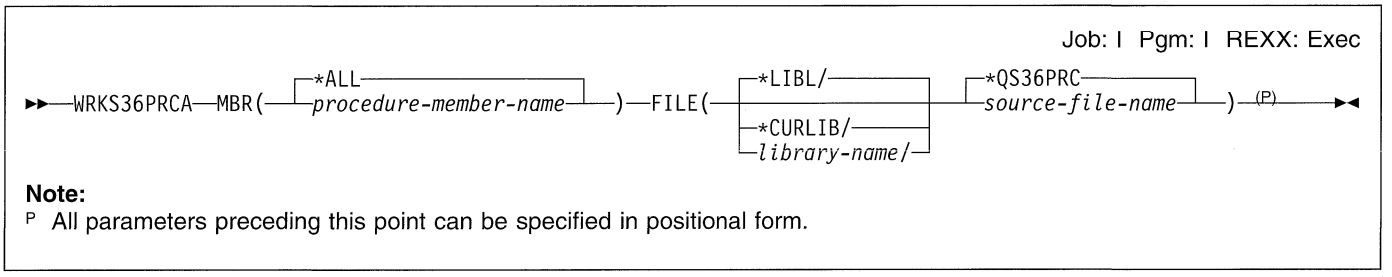
program-name: Specify a program name.

Example

```
WRKS36PGMA  PGM(IDENTIFY)
```

This command provides a list of programs from the program file IDENTIFY.

WRKS36PRCA (Work with System/36 Procedure Attributes) Command



Note:

| (P) All parameters preceding this point can be specified in positional form.

Purpose

The Work with System/36 Procedure Attributes (WRKS36PRCA) command shows the Work with System/36 Procedure Attributes display. The display is a list of procedures from a specified library.

Required Parameters

MBR

Specifies the name of a procedure member file.

***ALL:** The attributes of all procedure members in the file are specified.

procedure-member-name: Specify a procedure member name.

FILE

Specifies the name of the file being used by the program to which this command is applied.

| The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

***QS36PRC:** The default source physical file is used.

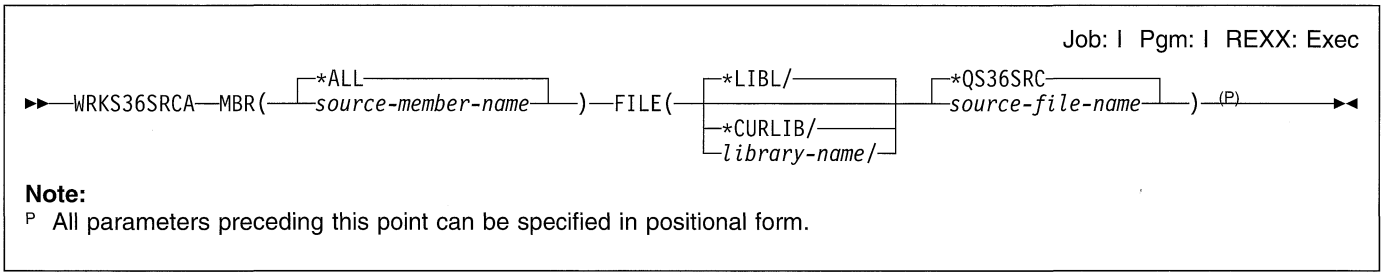
source-file-name: Specify the name of the source file.

Example

```
WRKS36PRCA  MBR(RESEARCH)  FILE(DOCUMENT)
```

This command provides a list of procedures from member file RESEARCH and the specified physical file DOCUMENT.

WRKS36SRCA (Work with System/36 Source Attributes) Command



Note:

| P All parameters preceding this point can be specified in positional form.

Purpose

The Work with System/36 Source Attributes (WRKS36PRCA) command shows the Work with System/36 Source Attributes display. The display is a list of source members within a specified source file.

Required Parameters

MBR

Specifies the name of a source member file.

***ALL:** The attributes of all procedure members in the file are specified.

procedure-member-name: Specify a procedure member name.

FILE

Specifies the name of the file being used by the program to which this command is applied.

| The name of the file can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

library-name: Specify the name of the library to be searched.

***QS36SRC:** The default source physical file is used.

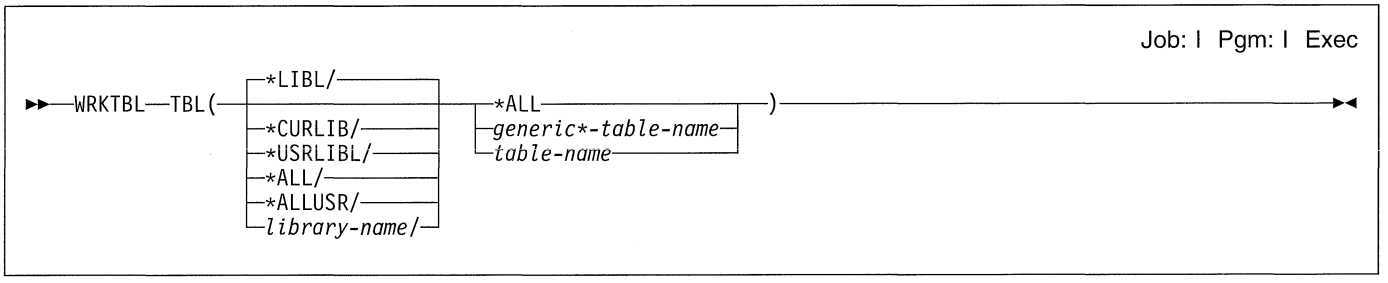
source-file-name: Specify the name of the source file.

Example

```
WRKS36SRCA  MBR(TEXT)  FILE(MESSAGE)
```

This command provides a list of source attributes from member file TEXT and the specified physical file MESSAGE.

WRKTBL (Work with Tables) Command



Purpose

The Work with Tables (WRKTBL) command allows you to show a list of tables from one or more libraries.

Restrictions:

1. Only the libraries to which you have USE authority will be searched.
2. Only the tables to which you have some authority will be shown on the display.
3. To perform operations on the tables, you must have USE authority to the command used by the operation, and the appropriate authority to the tables on which the operation is to be performed.

Required Parameters

TBL

Specifies a list of tables in the libraries that are shown. If no library qualifier is specified, *LIBL is assumed and all libraries in the job's library list are searched for the tables.

The name of the table can be qualified by one of the following library values:

***LIBL:** All libraries in the user and system portions of the job's library list are searched.

***CURLIB:** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used.

***USRLIBL:** Only the libraries in the user portion of the job's library list are searched.

***ALL:** All libraries in the system portion of the job's library list, including QSYS, are searched.

***ALLUSR:** All user libraries are searched. All libraries with names that do not begin with the letter Q are searched except for the following:

```
#CGULIB    #DFULIB    #RPGLIB    #SEULIB
#COBLIB    #DSULIB    #SDALIB
```

Although the following Qxxx libraries are provided by IBM, they typically contain user data that changes frequently. Therefore, these libraries are considered *user libraries*, and are also searched:

```
QDSNX      QPFRDATA   QUSER38
QGPL       QRCL       QUSRSYS
QGPL38     QS36F      QUSRVxRxMx
```

Note: A different library name, of the form QUSRVxRxMx, is added with each release. VxRxMx is the version, release, and modification level of the library.

library-name: Specify the name of the library to be searched.

***ALL:** All tables in the libraries identified in the library qualifier are shown.

generic-table-name:* Specify the generic name of the table. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

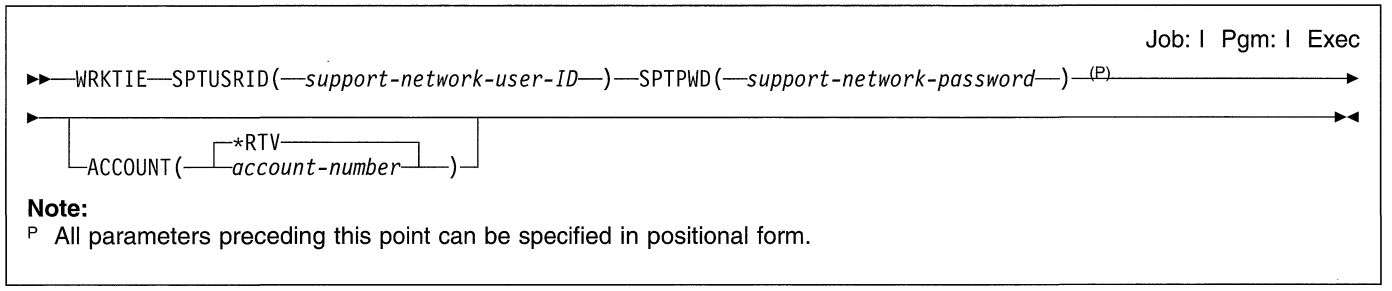
table-name: Specify the name of the tables that are shown.

Example

```
WRKTBL  TBL(LIB1/ABC*)
```

This command allows you to show a list of tables beginning with ABC that are in library LIB1.

WRKTIE (Work with Technical Information Exchange) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Technical Information Exchange (WRKTIE) command allows you to work with the TIE main menu where you can select TIE tasks. The TIE tasks are:

- Send TIE files
- Receive TIE files
- Query TIE files

Required Parameters

SPTUSRID

Specifies the user ID needed to sign on the remote support network.

SPTPWD

Specifies the password needed to sign on the remote support network.

Optional Parameters

ACCOUNT

Specifies the network account number used to sign on the remote support network. If the account number is not specified, the account number from the contact database is used.

***RTV:** The account number from the contact database is used (refer to the WRKCNTINF command).

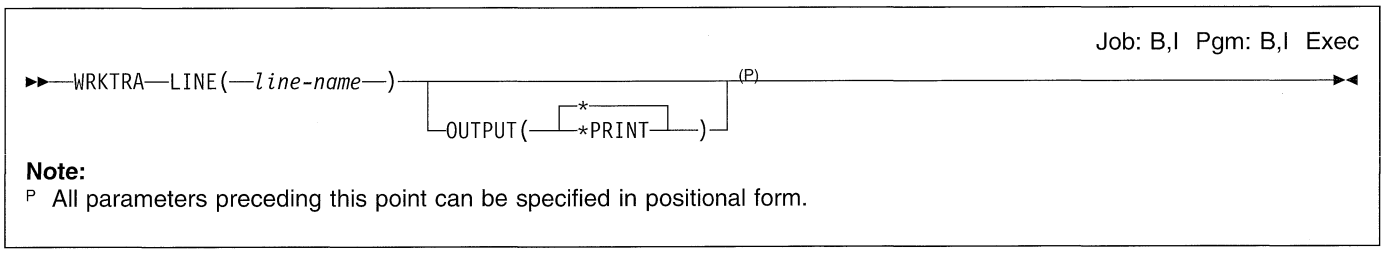
account-number: Specify the account number that is used to sign on the remote support network.

Example

```
WRKTIE  SPTUSRID(ACME)  SPTPWD(11111)
  ACCOUNT(11420880)
```

This command allows a user whose ID is ACME, whose password is 11111, and whose account number is 11420880 to work with the TIE main menu.

WRKTRA (Work with TRLAN Adapters) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with Token-Ring Local Area Network Adapters (WRKTRA) command allows the user to see and work with a list of active TRLAN (token-ring local area network) adapters.

Required Parameters

LINE
Specifies the name of the line that is attached to the adapters to be displayed.

Optional Parameters

OUTPUT
Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

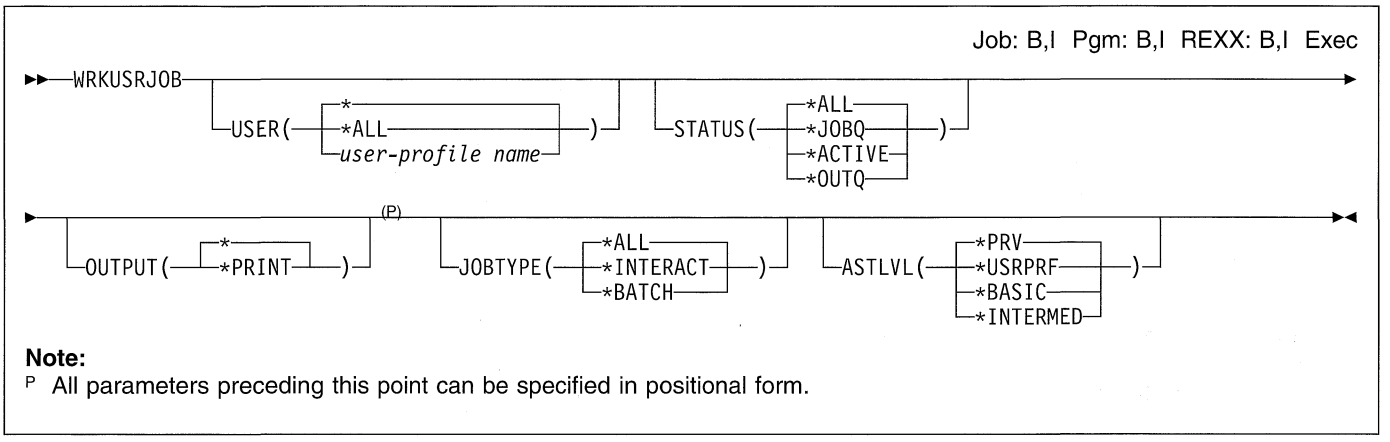
***PRINT:** The output is printed with the job's spooled output.

Example

```
WRKTRA  LINE(DETBRANCH)
```

This command allows the user to display and work with a list of adapters that are connected to the DETBRANCH line.

WRKUSRJOB (Work with User Jobs) Command



Note:

P All parameters preceding this point can be specified in positional form.

Purpose

The Work with User Jobs (WRKUSRJOB) command allows you to work with a list of selected user jobs.

Depending on the values you specify on the STATUS, ASTLVL, and JOBTYPE parameters, you can use this command to perform the following tasks:

- From the Work with Signed-On Users display, you can select options to send messages to or sign off users who are signed on the system. You can also select options to display messages or display details about signed-on users. To show the Work with Signed-On Users display, specify STATUS(*ACTIVE), ASTLVL(*BASIC), and JOBTYPE(*INTERACT).
- From the Work with Jobs display, you can show the status of all batch jobs running on the system. You can select options to hold, end, or release the jobs shown. You can also select options to work with printer output or display messages. To show the Work with Jobs display, specify any value for the STATUS parameter, ASTLVL(*BASIC), and JOBTYPE(*BATCH).
- From the Work with User Jobs display, you can show the status of user jobs running on the system and of user jobs that are on job queues or output queues. You can select options to change, hold, end, work with, release, or disconnect the jobs shown. You can also select options to work with spooled files and display messages. To show the Work with User Jobs display, specify one of the following:
  - ASTLVL(*INTERMED) and any value for the STATUS and JOBTYPE parameters
  - ASTLVL(*BASIC), JOBTYPE(*INTERACT), and either STATUS(*ALL), STATUS(*JOBQ), or STATUS(*OUTQ)
  - ASTLVL(*BASIC), JOBTYPE(*ALL), and any value for the STATUS parameter

Optional Parameters

USER

This parameter allows you to work with user jobs defined by the user profile of the job. User jobs include interactive jobs, submitted batch jobs, communications-evoked batch jobs, MRT batch jobs, and autostart jobs. User jobs do not include system jobs, subsystem monitor jobs, spooling readers, or spooling writers.

***:** User jobs with the current user profile are worked with.

***ALL:** User jobs with all user profiles are worked with.

user-profile name: Specify the name of a user profile. User jobs with the specified user profile are displayed.

STATUS

Specifies the status of the user jobs being worked with.

***ALL:** All statuses of user jobs can be worked with, including jobs on job queues, active jobs, and jobs on output queues.

***JOBQ:** Only user jobs that are on job queues are worked with.

***ACTIVE:** Only user jobs that are active are worked with, including all group jobs and system request jobs. You must specify STATUS(*ACTIVE) to show the Work with Signed-On Users display.

***OUTQ:** Only user jobs that have completed running but still have output on an output queue are worked with.

OUTPUT

Specifies whether the output is displayed at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

JOBTYPE
Specifies the type of the user jobs shown.

***ALL:** All types of user jobs are shown, including interactive jobs and batch jobs. If JOBTYPE(*ALL) is specified, the ASTLVL parameter is ignored.

***INTERACT:** Only interactive user jobs are shown. You must specify JOBTYPE(*INTERACT) to show the Work with Signed-On Users display. If you also specify ASTLVL(*BASIC), interactive jobs shown include suspended group jobs and signed-off users with printer output waiting to print.

***BATCH:** Only batch user jobs are shown, including prestart jobs, batch immediate jobs, and 36EE MRT jobs. You must specify JOBTYPE(*BATCH) to show the Work with Jobs display.

ASTLVL
Specifies the user interface to use.

***PRV:** The previous user interface is used.

***USRPRF:** The user interface specified in the user profile is used. If *ADVANCED is specified in the user profile, *INTERMED is used.

***BASIC:** The Operational Assistant user interface is used.

***INTERMED:** The system interface is used.

Examples

Example 1: Working with a List of Selected Jobs

```
WRKUSRJOB  USER(DICK)
```

This command allows the user to work with a list of jobs that are running with the user profile name DICK.

Example 2: Working with a List of All Jobs

```
WRKUSRJOB  USER(*ALL)  STATUS(*JOBQ)
```

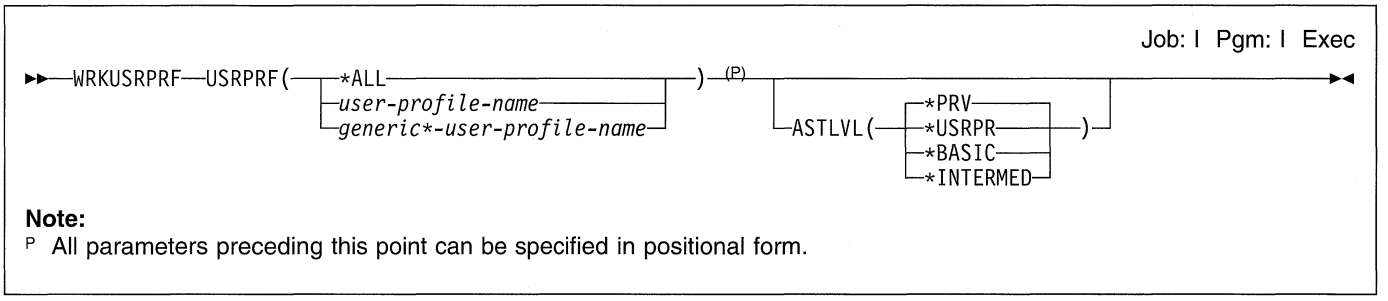
This command allows the user to work with a list of all the jobs on job queues.

Example 3: Working with a List of Interactive Jobs

```
WRKUSRJOB  USER(*ALL) STATUS(*ACTIVE)
  JOBTYPE(*INTERACT) ASTLVL(*BASIC)
```

This command allows the user to work with a list of signed-on users that includes signed-off users with printer output and suspended interactive group jobs.

WRKUSRPRF (Work with User Profiles) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with User Profiles (WRKUSRPRF) command allows you to work with a list of user profiles from which you can perform several related functions.

Restrictions:

1. Only the user profiles to which you have some authority will be shown on the display.
2. To perform operations on the user profiles, you must have USE authority to the command used by the operation, and the appropriate authority to the user profiles on which the operation is to be performed.

Required Parameters

USRPRF

Specifies the name or generic name of the user profiles you want to work with.

***ALL:** A list of all the user profiles that you own or have authority to see are shown.

user-profile-name: Specify the name of the user profile.

generic-user-profile-name:* Specify the generic name of the user-profile. A generic name is a character string of one or more characters followed by an asterisk (*); for example, ABC*. The asterisk substitutes for any valid characters. A generic name specifies all objects with names that begin with the generic prefix for which the user has authority. If an asterisk is not included with the generic (prefix) name, the system assumes it to be the complete object name. If the complete object name is specified, and multiple libraries are searched, multiple objects can be worked with only if *ALL or *ALLUSR library values can be specified for the name. For more information on the use of generic functions, refer to "Rules for Specifying Names."

Optional Parameters

ASTLVL

Specifies the user interface to use.

***PRV:** The previous user interface is used.

***USRPRF:** The user interface specified in the user profile is used.

***BASIC:** The Operational Assistant user interface is used.

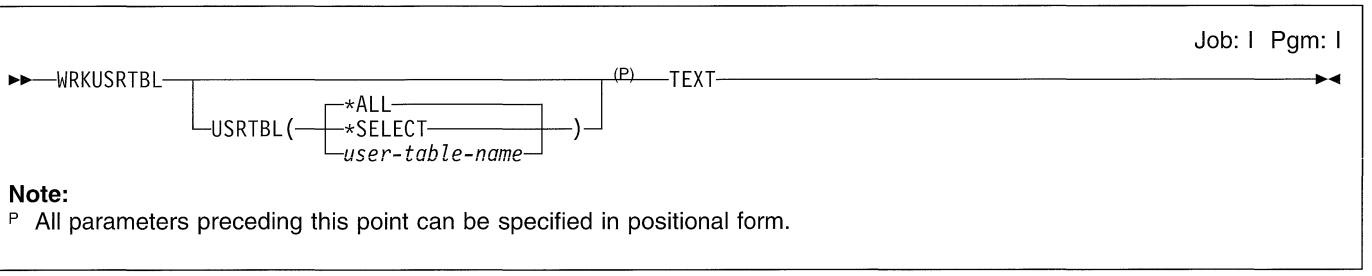
***INTERMED:** The system interface is used.

Example

```
WRKUSRPRF   (RO*)
```

This command allows you to work with a list of user profiles starting with the letters RO.

WRKUSRTBL (Work with User Tables) Command



Note:
P All parameters preceding this point can be specified in positional form.

Purpose

The Work with User Tables (WRKUSRTBL) command allows you to work with finance user tables. Once they are created, you can add or delete user IDs in the tables. Several finance user tables can be defined. Each table must have a unique name.

Finance user table updates can be accessed by any finance job that is submitted after all changes are completed.

Restriction: This command is shipped with public *EXCLUDE authority.

Optional Parameters

USRTBL

Specifies the name of a table that contains finance user IDs.

***ALL:** The list of existing user table entries is shown. On this display, you can create, change, delete, or display user tables.

***SELECT:** The list of existing user table entries is shown. This value, which performs the same function as *ALL, is included for compatibility with previous releases.

user-table-name: Specify the name of the user table with which to work.

TEXT

Unused parameter provided for compatibility with previous releases.

Examples

Example 1: Working With All Finance User Tables

```
WRKUSRTBL  USRTBL(*SELECT)
```

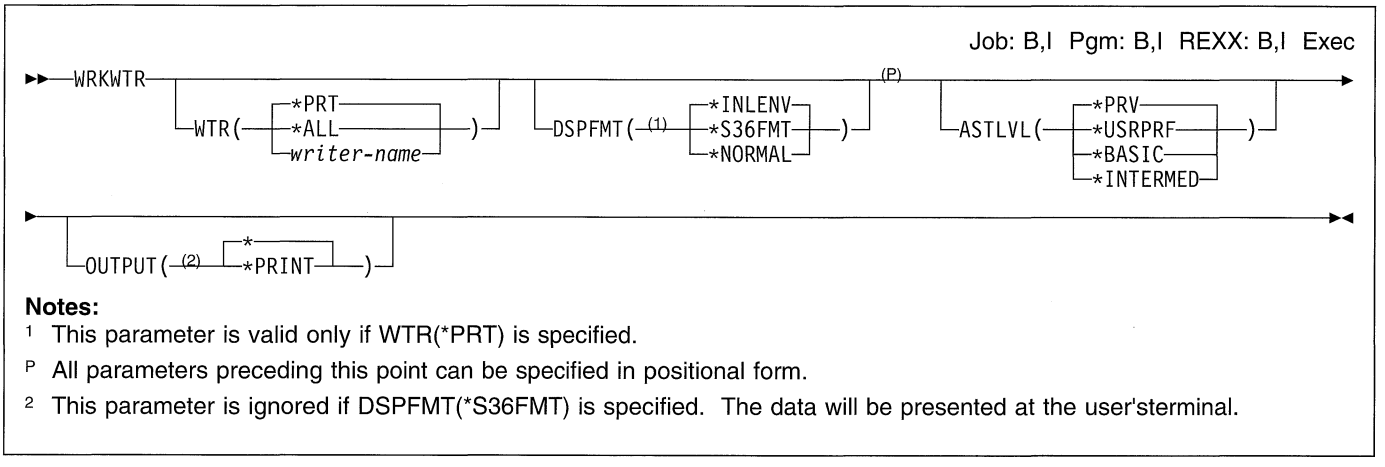
This command allows you to work with the names of all finance user tables. You can create a new table, select an existing table for update, or delete or display tables on the Work with User Table display.

Example 2: Working With One User Table

```
WRKUSRTBL  USRTBL(USRTBL1)
```

This command allows you to work with the user table USRTBL1. From this display, you can create, change, delete, or display a table.

WRKWTR (Work with Writers) Command



Notes:

1 This parameter is valid only if WTR(*PRT) is specified.

P All parameters preceding this point can be specified in positional form.

2 This parameter is ignored if DSPFMT(*S36FMT) is specified. The data will be presented at the user'sterminal.

Purpose

The Work with Writers (WRKWTR) command allows you to work with the status of printers and writers. This can be the overall status of all writers or all printers along with writer information for the printers, or detailed status of a specific writer. This status information may be directed to a display station or a printer. The status of the writers may change while the command is running.

Optional Parameters

WTR

Specifies the name of the spooling writer for which detailed information is displayed, or specifies that the main attributes and status of all spooling writers or all printers are shown.

***PRT:** All printers configured on the system are shown along with information about the writers started for them.

***ALL:** The attributes and the current status of all spooling writers are shown.

writer-name: Specify the name of the spooling writer for which the detailed description is shown.

DSPFMT

Specifies the format and terminology being used on the displays that result from entering this command.

***INLENV:** The format and terminology used is determined by the special environment value (SPCENV) specified in your user profile. If the SPCENV value is *S36, the System/36 terminology is used. All other values result in AS/400 system terminology being used.

***S36FMT:** System/36 terminology is used to present the information.

***NORMAL:** AS/400 system terminology is used to present the information.

ASTLVL

Specifies which user interface to use.

***PRV:** The previous user interface is used.

***USRPRF:** The user interface specified in the user profile is used.

***BASIC:** The Operational Assistant user interface is used.

***INTERMED:** The system interface is used.

OUTPUT

Specifies whether the output from the command is shown at the requesting work station or printed with the job's spooled output. More information on this parameter is in Appendix A, "Expanded Parameter Descriptions."

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

***PRINT:** The output is printed with the job's spooled output.

Examples

Example 1: Working With the Status of All Printers and Writers

```
WRKWTR  WTR(*ALL)
```

This command allows the user to work with the status of all printers and writers.

Example 2: Working With the Status of One Writer

```
WRKWTR  WTR(DISKWRITE)
```

This command allows the user to work with detailed information about writer DISKWRITE.

Part 4. Appendixes

Appendix A. Expanded Parameter Descriptions

This appendix contains the expanded descriptions of some of the parameters commonly used in the CL commands. The parameters included in this appendix meet one or both of these criteria:

- There is extensive information about how they are used.
- They are used in many of the CL commands (such as the AUT parameter), and the parameter description in the individual command description gives only the essential information.

The expanded descriptions of the applicable command parameters have been placed in this appendix to:

- Reduce the amount of material needed in the individual commands. Normally programmers familiar with a parameter's main function do not need the details.
- Provide the supplemental information that is useful to programmers in some instances.

The format for this appendix is designed for easy reference.

- The common parameters are organized in alphabetic order by their keywords.
- The name that identifies the first new parameter on each page is placed in the outer left margin of each page.
- A general description of each parameter is given that explains its function, states the rules for its use, and provides other helpful information.
- The values that can be specified for each parameter are listed. Each value is followed by an explanation of what it means and (possibly) in which commands it is used. Not all of the values appear in every command. Refer to the individual command descriptions for the specific use of the value in that command parameter.

AUT Parameter

The authority (AUT) parameter is used in create, grant, and revoke commands. It specifies the authority granted to all users of an object. It also specifies an authorization list that is used to secure the object. Four object types allow the AUT parameter to contain an authorization list: LIB, PGM, DTADCT, and FILE. Public authority is an OS/400 object attribute that controls the base set of rights to that object for all users having access to the system. These rights can be extended or reduced for specific users. If an authorization list is specified, the public authority in the authorization list is the public authority for the object. The owner of an object has all authority to the object at its creation.

If the object is created as a private object or with the limited authority given to all users, the owner can grant more or less authority to specific users by specifically naming them and stating their authority in the Grant Object Authority (GRTOBJAUT) command. The owner also can withdraw specific authority from specific users, or from all users (publicly authorized and/or specifically authorized) by using the Revoke Object Authority (RVKOBJAUT) command or the Edit Object Authority (EDTOBJAUT) command.

The *Security Reference* manual has a complete description of security provisions and applicable rights of use by object type.

Values Allowed

**LIBCRTAUT:* Public authority for the object is taken from the CRTAUT keyword of the library that is to contain the object. This value is determined when the object is created. If the CRTAUT value for the library changes after the object is created, the new value does not affect existing objects.

**USE: Use* authority allows you to perform basic operations on the object, such as running a program or reading a file. You are prevented from changing the object. Use authority provides object operational authority and read authority.

**CHANGE: Change* authority allows you to perform all operations on the object (except those limited to the owner or controller) by granting object existence authority and object management authority. You can change the object and perform basic functions on the object. Change authority provides object operational authority and all data authority.

**ALL: All* authority allows you to perform all operations on the object except those controlled by authorization list management authority. You can control the object's existence, specify the security for it, change it, and perform basic functions on it. You cannot transfer ownership of the object.

**EXCLUDE: Exclude* authority prevents you from accessing the object.

authorization-list-name: Specify the name of the authorization list whose authority is used.

CLS Parameter

The class (CLS) parameter identifies the attributes that define the run time environment of a job. The following attributes are defined in each class:

- Run priority: A number that specifies the priority level assigned to all jobs running that use the class. The priority level is used to determine which job, of all the jobs competing for system resources, is run next. The value can be 1 through 99, where 1 is the highest priority (all jobs having a 1 priority are run first).
- Time slice: The maximum amount of processor time that the system allows the job to run when it is allowed to begin. The time slice indicates the amount of time needed for the job to accomplish a meaningful amount of work (the time used by the system for reading auxil-

iary storage is not charged against the time slice). When the time slice ends, the job waits while other queued jobs of the same or higher priority are allowed to run (up to the time specified in their time slices); then the job is given another time slice.

- Purge value: Indicates whether the job step is eligible to be moved from main storage to auxiliary storage while the job is waiting for some resource before it can continue, or when its time slice is used up and equal or higher priority jobs are waiting.
- Default wait time: The default amount of time that the system waits for the completion of an instruction that performs a wait. This wait time applies to times when an instruction is waiting for a system action, not to the time an instruction is waiting for a response from a user. Normally, this would be the amount of time you are willing to wait for the system before ending the request. If the wait time is exceeded, an error message is passed to the job. This default wait time applies only when a wait time is not specified in the CL command that causes the wait.

  The wait time used for allocating file resources is specified in the file description and can be overridden by an override command. It specifies that the wait time specified in the class object is used. If file resources are not available when the file is opened, the system waits for them until the wait time ends.

  Note: The class attributes apply to each routing step of a job. Most jobs have only one routing step, but if the job is rerouted (because of something like the Remote Job or Transfer Job command) the class attributes will be reset.
- Maximum CPU time: The maximum amount of processor time (the sum of the time slices if they differ, or time slice period multiplied by number of time slices if they are equal) allowed for a job's routing step to complete processing. If the job's routing step is not completed in this amount of time, it is ended, and a message is written to the job log.
- Maximum temporary storage: The maximum amount of temporary storage that can be used by a job's routing step. This temporary storage is used for the programs that run in the job, for the system objects used to support the job, and for temporary objects created by the job.

The system is shipped with a set of classes that define the attributes for several job processing environments. Other classes can be created by the Create Class (CRTCLS) command; any class can be displayed or deleted by the respective Display Class (DSPCLS) and Delete Class (DLTCLS) commands.

The following classes (by name) are supplied with the system:

QGPL/QBATCH	For use by batch jobs
QSYS/QCTL	For use by the controlling subsystem
QGPL/QINTER	For use by interactive jobs
QGPL/QPGMR	For use by the programming subsystem
QGPL/QSPL	For use by the spooling subsystem printer writer
QGPL/QSPL2	For general spooling use in the base system pool

Values Allowed

qualified-class-name: Specify the name of the class, optionally qualified by the name of the library in which the class is stored. If the class name is not qualified and the CLS parameter is in the CRTCLS command, the class object is stored in *CURLIB; otherwise, the library list (*LIBL) is used to find the class name.

COUNTRY Parameter

The Country parameter specifies the country code part of the X.400 O/R name. An ISO 3166 Alpha-2 code or a CCITT country code can be specified. Table 1 is a list of the possible country codes that can be specified.

Values Allowed

**NONE:* No country code is specified.

country-code: Specify an ISO 3166 Alpha-2 code or a CCITT country code from the following table.

Table 78 (Page 1 of 3). ISO X.400 Country Code Table

Country	ISO 3166 Alpha-2 Code	CCITT Country Code
Afghanistan	AF	412
Albania	AL	276
Algeria	DZ	603
American Samoa	AS	544
Andorra	AD	
Angola	AO	631
Anguilla	AI	
Antarctica	AQ	
Antigua and Barbuda	AG	344
Argentina	AR	722
Aruba	AW	362
Australia	AU	505
Austria	AT	232
Bahamas	BS	364
Bahrain	BH	426
Bangladesh	BD	470
Barbados	BB	342
Belguim	BE	206
Belize	BZ	702
Benin	BJ	616
Bermuda	BM	350
Bhutan	BT	
Bolivia	BO	736

Table 78 (Page 2 of 3). ISO X.400 Country Code Table

Country	ISO 3166 Alpha-2 Code	CCITT Country Code
Botswana	BW	652
Bouvet Island	BV	
Brazil	BR	724
Br. Indian Ocean Terr.	IO	
Brunei Darussalam	BN	528
Bulgaria	BG	284
Burkina Faso	BF	613
Burma	BU	414
Burundi	BI	642
Byelorussian SSR	BY	
Cameroon	CM	624
Canada	CA	302
Cape Verde	CV	625
Cayman Islands	KY	346
Central African Republic	CF	623
Chad	TD	622
Chile	CL	730
China	CN	460
Christmas Island	CX	
Cocos (Keeling) Is.	CC	
Colombia	CO	732
Comoros	KM	654
Congo	CG	629
Cook Islands	CK	548
Costa Rica	CR	712
Cote D'Ivoire	CI	612
Cuba	CU	368
Cyprus	CY	280
Czechoslovakia	CS	230
Denmark	DK	238
Djibouti	DJ	638
Dominica	DM	366
Dominican Republic	DO	370
East Timor	TP	
Ecuador	EC	740
Egypt	EG	602
El Salvador	SV	706
Equatorial Guinea	GQ	627
Ethiopia	ET	636
Falkland Islands (Malvinas)	FK	
Faroe Islands	FO	288
Fiji	FJ	542
Finland	FI	244
France	FR	208
French Guiana	GF	742
French Polynesia	PF	547
French Southern Terr.	TF	
Gabon	GA	628
Gambia	GM	607
German Democratic Republic	DD	218
Germany, Federal Republic of	DE	262
Ghana	GH	620
Gibralter	GI	266
Greece	GR	202
Greenland	GL	290

Table 78 (Page 2 of 3). ISO X.400 Country Code Table

Country	ISO 3166 Alpha-2 Code	CCITT Country Code
Grenada	GD	352
Guadeloupe	GP	
Guam	GU	535
Guatemala	GT	704
Guinea	GN	611
Guinea-Bissau	GW	632
Guyana	GY	738
Haiti	HT	372
Heard and McDonald Islands	HM	
Honduras	HN	708
Hong Kong	HK	454
Hungary	HU	216
Iceland	IS	274
India	IN	404
Indonesia	ID	510
Iran	IR	432
Iraq	IQ	418
Ireland	IE	272
Israel	IL	425
Italy	IT	222
Jamaica	JM	338
Japan	JP	440
Jordan	JO	416
Kampuchea, Democratic	KH	456
Kenya	KE	639
Kiribati	KI	
Korea, Democratic People's Republic of	KP	467
Korea, Republic of	KR	450
Kuwait	KW	419
Lao People's Democratic Republic	LA	457
Lebanon	LB	415
Lesotho	LS	651
Liberia	LR	618
Libyan Arab Jamahiriya	LY	606
Liechtenstein	LI	
Luxembourg	LU	270
Macau (Macao)	MO	455
Madagascar	MG	646
Malawi	MW	650
Malaysia	MY	502
Maldives	MV	472
Mali	ML	610
Malta	MT	278
Marshall Islands	MH	
Martinique	MQ	
Mauritania	MR	609
Mauritius	MU	617
Mexico	MX	334
Micronesia	FM	
Monaco	MC	212
Mongolia	MN	428
Montserrat	MS	354
Morocco	MA	604

Table 78 (Page 3 of 3). ISO X.400 Country Code Table

Country	ISO 3166 Alpha-2 Code	CCITT Country Code
Mozambique	MZ	643
Namibia	NA	649
Nauru	NR	536
Nepal	NP	429
Netherlands	NL	204
Netherlands Antilles	AN	362
Neutral Zone	NT	
New Caledonia	NC	546
New Zealand	NZ	530
Nicaragua	NI	710
Niger	NE	614
Nigeria	NG	621
Niue	NU	
Norfolk Island	NF	
Northern Mariana Islands	MP	
Norway	NO	242
Oman	OM	422
Pakistan	PK	410
Palau	PW	
Panama	PA	714
Papua New Guinea	PG	537
Paraguay	PY	744
Peru	PE	716
Philippines	PH	515
Pitcairn	PN	
Poland	PL	260
Portugal	PT	268
Puerto Rico	PR	330
Qatar	QA	427
Reunion	RE	647
Romania	RO	226
Rwanda	RW	635
St. Helena	SH	
St. Kitts and Nevis	KN	356
St. Lucia	LC	358
St. Pierre and Miquelon	PM	308
St. Vincent and the Grenadines	VC	360
Samoa	WS	549
San Marino	SM	
Sao Tome and Principe	ST	626
Saudi Arabia	SA	420
Senegal	SN	608
Seychelles	SC	633
Sierra Leone	SL	619
Singapore	SG	525
Solomon Islands	SB	540
Somalia	SO	637
South Africa	ZA	655
Spain	ES	214
Sri Lanka	LK	413
Sudan	SD	634
Suriname	SR	746
Svalbard and Jan Mayen Is.	SJ	
Swaziland	SZ	653
Sweden	SE	240

Table 78 (Page 3 of 3). ISO X.400 Country Code Table

Country	ISO 3166 Alpha-2 Code	CCITT Country Code
Switzerland	CH	228
Syrian Arab Republic	SY	417
Taiwan, Province of China	TW	
Tanzania, United Republic of	TZ	640
Thailand	TH	520
Togo	TG	615
Tokelau	TK	
Tonga	TO	539
Trinidad and Tobago	TT	374
Tunisia	TN	605
Turkey	TR	286
Turks and Caicos Is.	TC	376
Tuvalu	TV	
Uganda	UG	641
Ukrainian SSR	UA	
United Arab Emirates	AE	424
United Arab Emirates (Abu Dhabi)	AE	430
United Arab Emirates (Dubai)	AE	431
United Kingdom	GB	234, 235
United States	US	310, 311, 312, 313, 314, 315, 316
United States Minor Outlying Is.	UM	
Uruguay	UY	748
USSR	SU	250
Vanuatu	VU	
Vatican City State (Holy See)	VA	
Venezuela	VE	734
Viet Nam	VN	452
Virgin Is. (Brit.)	VG	348
Virgin Is. (U.S.)	VI	332
Wallis and Funtuna Is.	WF	543
Western Sahara	EH	
Yemen	YE	421
Yemin, Democratic	YD	423
Countries of the former Yugoslavia	YU	220
Zaire	ZR	630
Zambia	ZM	645
Zimbabwe	ZW	648

EXCHTYPE Parameter

The Exchange Type (EXCHTYPE) parameter specifies which one of the three diskette exchange types (basic, H, or I) is used when the system writes diskette files. The exchange type is stored in the volume label area on the diskette. There is one label for each data file on the diskette.

The exchange type used can be specified in one of the diskette device file commands (Create Diskette File (CRTDKTF), Change Diskette File (CHGDKTF), or Override Diskette File (OVRDKTF)), or it can be passed as a param-

eter when the device file is opened by the high-level language program (if supported).

The exchange type specified in the diskette device file or high-level language program is not used by the system when processing a diskette input file. Instead, the system uses the exchange type from the file label on the diskette.

It is possible for one diskette to contain both basic and I exchange format files or H and I format files, but it is not possible for one diskette to contain both basic and H format files.

Values Allowed

One of the following values can be specified for the EXCHTYPE parameter, depending on the diskette type (1, 2, or 2D) and the diskette sector size (128, 256, 512, or 1024 bytes):

**STD:* The exchange type is determined by the system, and it is based on the diskette type and sector size. A basic exchange type is used if the diskette type is 1 or 2, and the diskette sector size is 128 bytes. An H exchange type is used if the diskette type is 2D and the diskette sector size is 256 bytes. The *STD value is not valid for any other combination of type and sector size.

**Basic:* The basic exchange type is used. The diskette type must be 1 or 2, and the diskette sector size must be 128 bytes.

**H:* The H exchange type is used. The diskette type must be 2D, and the diskette sector size must be 256 bytes.

**I:* The I exchange type is used. The diskette type and sector size may be any of the following:

Diskette Type	Sector Size (bytes)
1	128, 256, or 512
2	128, 256, or 512
2D	256, 512, or 1024

FILETYPE Parameter

The FILETYPE parameter specifies whether the database file description describes data records or source records. Further, it specifies whether each member of a database file being created is to contain data records or source records (statements). For example, the file could contain RPG source statements for an RPG program or data description source (DDS) statements for another device or database file.

Note: If you are creating a source type *physical* database file and are not providing field-level descriptions of the file (through data description specifications (DDS)), you can use either the Create Physical File (CRTPF) command or the Create Source Physical File (CRTSRCPF) command. However, the CRTSRCPF command is usually more convenient and efficient, because it is designed to be used to create source physical files. If DDS is provided when you are creating a source type database file, you should use the CRTPF command or the Create Logical File (CRTLF) command, which both have the SRCFILE and SRCMBR parameters for specifying source input.

Records in a source file must have at least three fields: the first two are the source sequence number field and the date field; the third field contains the source statement. These three fields are automatically provided by the OS/400 when a source file is created for which no DDS is provided; additional source fields can be defined in DDS. The length of the sequence number field must be six zoned digits with two decimal places. The length of the date field must be six zoned digits with no decimal places.

The source sequence number and date fields are added to the source record when:

- Records are read into the system.
- Records are created by the Source Entry Utility (which is part of the licensed Application Development Tools program).

The fields are added when an inline data file (specified as the standard source file format) is read from the diskette device. The spooling reader places a sequence number in the source sequence number field and sets up a zeroed date field.

If those fields already exist in records read from the diskette device, they are not changed. If the records in a database file are in source format and are being read as an inline data file in data format, the source sequence number and date fields are removed.

More information about data and source files is in the *Database Guide*.

Values Allowed

**DATA:* The file created contains or describes data records.

**SRC:* The file created contains or describes source records. If the file is keyed, the 6-digit source sequence number field must be used as the key field.

FRCRATIO Parameter

The force write ratio (FRCRATIO) parameter specifies the maximum number of records that can be inserted, updated, or deleted before they are forced into auxiliary (permanent) storage. The force write ratio ensures that all inserted, updated, or deleted records are written into auxiliary storage at least as often as this parameter specifies. In the event of system failure, the only records likely to be lost would be those that were inserted, updated, or deleted since the last force write operation.

The force write ratio is applied to all records inserted, updated, or deleted in the file through the open data path (ODP) to which the force write ratio applies. If two programs are sharing the file, SHARE(*YES), the force write ratio is not applied separately to the set of records inserted, updated, or deleted by each program, but is applied to any combination of records (from both programs) that equals the specified force write ratio parameter value. For example, if a force write ratio of 5 was specified for the file, any combination of five records from the two programs (such as four from one program and one from the other) forces the records to be written to auxiliary storage. If two or more programs are using the file through separate ODPs, the insertions, updates, and deletions from each program are accumulated individually for each ODP.

Each database file can have a force write ratio assigned to it. Logical files, which can access data from more than one physical file, can specify a more restrictive force write ratio (a smaller number of records) than that specified for the based-on physical files. However, a logical file cannot specify a *less* restrictive force write ratio. If a logical file specifies a less restrictive force write ratio than that specified for any of the physical files, the most restrictive force write ratio from the physical files is used for the logical file. For example, if the force write ratios of three physical files are 2, 6, and 8, the force write ratio of a logical file based on these physical files cannot be greater than 2. If no force write ratio is specified for the logical file, 2 is assumed. Thus, each time a program inserts, updates, or deletes two records in the logical file (regardless of which physical files are affected), those records are forced into auxiliary storage.

The FRCRATIO number overrides the SEQONLY number specified. For example, if you specify:

```
OVRDBF ... SEQONLY(*YES 20) FRCRATIO(5)
```

The value of 20 is overridden and a buffer of five records is used. When FRCRATIO(1) is used, a buffer still exists, but it contains only a single record.

Access paths associated with the inserted, updated, and deleted records are written to auxiliary storage only when all the records covered by the access path have been written to auxiliary storage. If only one ODP exists for the file, the access path is forced to auxiliary storage whenever a forced write occurs. If two or more ODPs to the file exist, the access path is written to auxiliary storage whenever all the inserted, updated, and deleted records for all the ODPs have been forced.

Notes:

1. These rules apply only when a force write ratio of 2 or higher is specified. When a force write ratio of 1 is specified, the access path is not written to auxiliary storage until all the ODPs have been closed.
2. If the file is being recorded in a journal. FRCRATIO(*NONE) should be specified. More information is in the *Advanced Backup and Recovery Guide*.

Values Allowed

**NONE:* There is no specified ratio; the system determines when the records are written to auxiliary storage.

number-of-records-before-force: Specify the number of updated, inserted, or deleted records that are processed before they are explicitly forced to auxiliary storage.

IGCFEAT Parameter

The IGCFEAT parameter specifies which double-byte character set (DBCS) table is used, according to device and language. The figure below indicates the corresponding IGCFEAT parameter and DBCS font table for the double-byte character set device being configured.

Table 79 (Page 1 of 2). DBCS Features Configurable on the IGCFEAT Parameter

Language/Device	Type of Physical DBCS Work Station	Configure as Type-Model	Configure with DBCS Feature
Japanese Display Stations	5295-001 Display	5555-B01	((2424J4 55FE))
	5295-002 Display	5555-B01	((2424J4 68FE))
	PS/55* running 5250PC	5555-B01	((2424J4 68FE))
	PS/55 running 5250PC/2	5555-E01	((2424J0 (1)))
Japanese 24x24 Printers	Attached to 5295-001 Displays	5553-B01	((2424J1 55FE))
	Attached to 5295-002 Displays	5553-B01	((2424J1 68FE))
	Attached to PS/55	5553-B01	((2424J1 68FE))
	5227-001 Printer	5553-B01	((2424J2 55FE))
	5327-001 Printer	5553-B01	((2424J1 68FE))
Japanese 32x32 Printers	5337-001 Printer	5553-B01	((3232J0 (1)))
	5383-200 Printer	5583-200	((3232J0 (1)))
Korean Display Stations	All display stations	5555-B01	((2424K0 (1)))
Korean 24x24 Printers	Attached to 5295 Displays	5553-B01	((2424K0 (1)))
	Attached to PS/55	5553-B01	((2424K0 (1)))
	5227-002 Printer	5553-B01	((2424K2 52FE))
Traditional Chinese Display Stations	All display stations	5555-B01	((2424C0))

Table 79 (Page 2 of 2). DBCS Features Configurable on the IGCFEAT Parameter

Language/Device	Type of Physical DBCS Work Station	Configure as Type-Model	Configure with DBCS Feature
Traditional Chinese 24x24 Printers	Attached to 5295 Displays	5553-B01	((2424C0))
	Attached to PS/55	5553-B01	((2424C0))
	5227-003 Printer	5553-B01	((2424C2 5CFE))
Simplified Chinese Display Stations	All display stations	5555-B01	((2424S0))
Simplified Chinese 24x24 Printers	Attached to PS/55	5553-B01	((2424S0))
	5227-005 Printer	5553-B01	((2424S2 6FFE))

Using Double-Byte Character Text in CL Commands: You can use double-byte character data anywhere in a CL command that descriptive text can be used.

Enter double-byte character text as follows:

1. Begin the double-byte character text with an apostrophe (').
2. Enter a shift-out character.
3. Enter the double-byte character text.
4. Enter a shift-in character.
5. End the double-byte character text with an apostrophe (').

For example, to enter the double-byte character literal ABC, enter the following, where 0_E represents the shift-out character and 0_F represents the shift-in character:

'0_EABC0_F'

Limit the length of a double-byte character text description of an object to 14 double-byte characters, plus the shift control characters, to make sure that the description is properly displayed and printed.

JOB Parameter

The JOB parameter specifies the name of the job to which the command is applied. The job name identifies all types of jobs on the system. Each job is identified by a qualified job name, which has the following format:

job-number/user-name/job-name

Note that, although the syntax is similar, job names are qualified differently than OS/400 object names.

- Job number: The job number is a unique 6-digit number that is assigned to each job by the system. The job number provides a unique qualifier if the job name is not otherwise unique. The job number can be determined by means of the Display Job (DSPJOB) command. If specified, the job number must have exactly six digits.
- User name: The user name identifies the user profile under which the job is to run. The user name is the same as the name of the user profile and contains a maximum of 10 alphanumeric characters. The name can come from one of several sources, again, depending on the type of job:
  - Batch job: The user name is specified on the SBMJOB command, or it is specified in the job description referenced by the BCHJOB or SBMJOB commands.
  - Interactive job: The user name is specified at sign-on, or the user name is provided from the default in the job description referred to by the work station's job entry.
  - Autostart job: The user name is specified in the job description referred to by the job entry for the autostart job.
- Job name: The job name can contain a maximum of 10 alphanumeric characters, of which the first character must be alphabetic. The name can come from one of three sources, depending on the type of job:
  - Batch job: The job name is specified on the Batch Job (BCHJOB) or Submit Job (SBMJOB) commands or, if not specified there, the unqualified name of the job description is used.
  - Interactive job: The job name is the same as the name of the device (work station) from which the sign-on was performed.
  - Autostart job: The job name is provided in the autostart job entry in the subsystem description under which the job runs. The job name was specified in the Add Autostart Job Entry (ADDAJE) command.

Commands only require that the simple name be used to identify the job. However, additional qualification must be used if the simple job name is not unique.

Duplicate Job Names

If a duplicate job name is specified in a command in an *interactive* job, the system displays all of the duplicates of the specified job name to the user in qualified form. The job names are displayed in qualified form along with the user name and job number so that the user can further identify the job that is to be specified in a command. The correct qualified job name can then be entered.

If a duplicate job name is used in a command in a *batch* job, the command is not processed. Instead, an error message is written to the job log.

Values Allowed

The JOB parameter can have one or more of the following values, depending upon the command:

**:* The job is the one in which the command is entered; that is, the command with JOB(*) specified on it.

**JOBD:* The simple job name is the unqualified name of the job description.

**NONE:* No job name is specified as in the Display Log (DSPLOG) command.

job-name: A simple job name is specified.

qualified-job-name: A qualified job name is specified. If no job qualifier (user name and job number) is given, all of the jobs currently in the system are searched for the job name. If duplicates of the specified name are found, a qualified job name must be specified.

LABEL Parameter

The LABEL parameter specifies the data file identifier of the data file (on diskette or tape) used in input and/or output operations. The data file can be in either the exchange format or the save/restore format. Note, however, that the device file commands are used for diskettes and tapes that are in the exchange format only, *not* for those in the save/restore format; user-defined device files are not used in save/restore operations. Each data file that is on a diskette or tape has its data file identifier stored in its own file label.

On diskette, all of the data file labels are in one place, in the volume label area of that diskette. In addition to the data file identifier, each label contains other information about the file, such as where the file is stored on the diskette (track and sector) and whether the file continues on another diskette (in the case of a multivolume data file).

On tape, the data file label (or header label) of each data file is stored on the tape just before the data in the file. That is, each file on the tape has its own header label and its own data records together as a unit, and one file follows another. In addition to the data file identifier, each label also contains other information about the file, such as the file sequence number, record and block attributes, and whether it is a multivolume data file.

Generally, the data file identifier is an alphanumeric character string that contains no more than 8 characters. However, the maximum length actually depends on several things: what data format is used for the files, whether the files are on diskette or on tape, and CL commands in which the identifiers are specified. The unused portion of the file identifier field is left blank.

The first character of the data file identifier must be alphabetic (A through Z, $, #, or @) and the rest of the characters *should* be alphanumeric (A through Z, 0 through 9, $, #, _, ., and @). Special characters can be used if the identifier is enclosed in apostrophes. However, if the diskette or tape is used on a system other than an AS/400 system, the requirements for specifying identifiers on that system must be considered.

Diskette and Tape Data File Identifiers

For *diskettes* in the exchange format, the data file identifier cannot exceed 8 characters (the same limit as for RPG file names). This limitation applies to the following commands: Create Diskette File (CRTDKTF), Change Diskette File (CHGDKTF), Change Spooled File Attributes (CHGSPLFA), Override Diskette File (OVRDKTF), Delete Diskette Label (DLTDKTLBL), Display Diskette (DSPDKT), and Start Diskette Reader (STRDKTRDR).

For *tapes*, the identifier can have as many as 17 characters. However, if a tape is used on a system other than an AS/400 system, a maximum of 8 characters, or a qualified identifier of no more than 17 characters, should be used. If more than 8 characters are used, the identifier should be qualified and enclosed in apostrophes so that no more than 8 characters occur in either part, and the parts are separated by a period; for example, LABEL('TAXES.JAN1980'). This limitation applies to the following commands: Create Tape File (CRTTAPF), Change Tape File (CHGTAPF), Override Tape File (OVRTAPF), and Display Tape (DSPTAP).

Duplicate data file identifiers are not allowed in the same volume, on either diskette or tape. However, the identifier can be the same as the name of the database file written to diskette or tape if the file name contains no more than 8 characters. The diskette and tape data files contain only data, not file descriptions like those of database files. On diskette, the identifiers ERRORSET and SYSAREA cannot be used; they are reserved for special use.

The data file identifier is put on the volume when the data file is put on the volume. For input/output operations, the identifier can be specified in one of the diskette or tape device file commands, or it can be passed as a parameter when the device file is opened by the high-level language program that uses the file.

Save/Restore Format

For *diskettes* in the save/restore format, the data file identifier can have a maximum of 17 characters. If a library name is used to generate the label, the file identifier can have a maximum of 15 characters. The identifier consists of a library name of up to 10 characters followed by a period, a Q, and a 3-digit sequence number; for example, LABEL('PAYLIB.Q014'). The 15-character limit applies to identifiers of save/restore data files displayed by the DSPDKT command.

For *tapes* in the save/restore format, the identifier can have a maximum of 17 characters. If a library name is used to generate the label, the identifier cannot exceed 10 characters. You may specify a label other than a library name.

Values Allowed

One of the following values can be specified for the LABEL parameter, depending upon the command.

**ALL:* Labels for all the data file identifiers in the specified diskette or tape volumes are shown on the display.

**NONE:* The data file identifier is not specified. It must be supplied before the device file (and/or database file) is opened to be used in the diskette or tape operation.

**SAME:* The data file identifier already present in the diskette or tape device file does not change.

data-file-identifier: Specify the identifier of the data file (on diskette or tape) used or displayed with the device file description.

**LIB:* The file label is created by the system and the name of the library specified on the LIB parameter is used as the qualifier for the file name.

**SAVLIB:* The file label is created by the system, and the name of the library specified on the SAVLIB parameter is used as the qualifier for the file name.

MAXACT Parameter

The maximum activity level (MAXACT) parameter specifies the maximum number of jobs that can be concurrently started and that remain active through a job queue entry, communications entry, routing entry, or work station entry. A job is considered active from the time it starts running until it is completed. This includes time when:

- The job is actually being processed.
- The job is waiting for a response from a work station user.
- The job is started and available for processing but is not actually using the processor. For example, it might have used up its time slice and is waiting for another time slice.
- The job is started but is not available for processing. For example, it could be waiting for a message to arrive on its message queue.

Values Allowed

**NOMAX:* There is no limit to the number of jobs that can be concurrently active through this entry.

maximum-active-jobs: Specify a value that indicates the maximum number of jobs that can be concurrently active through this entry.

The *Programming: Work Management Guide* has a description of activity level controls.

OBJ Parameter

The object (OBJ) parameter specifies the names of one or more objects affected by the command in which this parameter is used. All of the objects must be in the library specified in the LIB parameter, the SAVLIB parameter, or the library qualifier in the OBJ parameter, depending upon which command is used.

On some commands, the generic name of a group of objects can be specified. To form a generic name, add an asterisk (*) after the last character in the common group of characters; for example, ABC*. If an * is not included with the name, the system assumes that the name is a complete object name.

Values Allowed

Depending on the command, the following types of values can be specified on the OBJ parameter:

- *ALL
- Simple object name
- Qualified object name
- Generic object name
- Qualified generic object name

OBJTYPE Parameter

The object type (OBJTYPE) parameter specifies the types of OS/400 objects that can be operated on by the command in which they are specified. The object types that can be specified in the OBJTYPE parameter vary from command to command.

The predefined values for all the OS/400 object types are listed below. When an object is created and a library qualifier can be specified but is not, the object is stored in the user's current job library, as shown in the last column. The user profile for each user specifies the user's current library. The current library will be QGPL if it is not specified otherwise. The other objects, identified by N/A in the last column, cannot be stored in user-provided libraries.

Table 80 (Page 1 of 2). Predefined Values and Default Library Locations for OS/400 Object Types

Value	Object Type	Default User Library
*ALRTBL	Alert table	*CURLIB
*AUTL	Authorization list	N/A
*BNDDIR	Binding directory	*CURLIB
*CFGL	Configuration list	N/A
*CHTFMT	Chart format	*CURLIB
*CLD	C description	*CURLIB
*CLS	Class	*CURLIB
*CMD	Command	*CURLIB

Table 80 (Page 2 of 2). Predefined Values and Default Library Locations for OS/400 Object Types

Value	Object Type	Default User Library
*CNNL	Connection list	QSYS
*COSD	Class-of-service description	N/A
*CSI	Communications Side Information	*CURLIB
*CSPMAP	Cross System Product map	*CURLIB
*CSPTBL	Cross System Product table	*CURLIB
*CTLD	Controller description	N/A
*DEVD	Device description	N/A
*DOC	Document	QDOC
*DTAARA	Data area	*CURLIB
*DTADCT	Data dictionary	N/A
*DTAQ	Data queue	*CURLIB
*EDTD	Edit description	N/A
*FCT	Forms control table	*CURLIB
*FILE	File	*CURLIB
*FLR	Folder	QDOC
*FNTRSC	Font resources	*CURLIB
*FORMDF	Form definition	*CURLIB
*FTR	Filter	*CURLIB
*GSS	Graphics symbol set	*CURLIB
*IGCDCT	Double-byte character set (DBCS) conversion dictionary	*CURLIB
*IGCSRT	Double-byte character set (DBCS) sort table	*CURLIB
*IGCTBL	Double-byte character set (DBCS) font table	N/A
*JOBD	Job description	*CURLIB
*JOBQ	Job queue	*CURLIB
*JOBSCD	Job schedule	*CURLIB
*JRN	Journal	*CURLIB
*JRNRCV	Journal receiver	*CURLIB
*LIB	Library	N/A
*LIND	Line description	N/A
*MENU	Menu description	*CURLIB
*MODD	Mode description	N/A
*MODULE	Compiler unit	*CURLIB
*MSGF	Message file	*CURLIB
*MSGQ	Message queue	*CURLIB
*NODL	Node list	*CURLIB
*NWID	Network interface description	QSYS
*OUTQ	Output queue	*CURLIB
*OVL	Overlay	*CURLIB
*PAGDFN	Page definition	*CURLIB
*PAGSEG	Page segment	*CURLIB
*PDG	Print Descriptor Group	*CURLIB

Table 80 (Page 2 of 2). Predefined Values and Default Library Locations for OS/400 Object Types

Value	Object Type	Default User Library
*PGM	Program	*CURLIB
*PNLGRP	Panel group definition	*CURLIB
*PRDAVL	Product availability	QSYS
*PRDDFN	Product definition	QSYS
*PRDLOD	Product load	QSYS
*QMFORM	Query management form	*CURLIB
*QMQRY	Query management query	*CURLIB
*QRYDFN	Query definition	QGPL
*RCT	Reference code translate table	QGPL
*SBSD	Subsystem description	*CURLIB
*SCHIDX	Information search index	QGPL
*SPADCT	Spelling aid dictionary	QGPL
*SRVPGM	Service program	*CURLIB
*SQLPKG	Structured Query Language package	*CURLIB
*SSND	Session description	QGPL
*S36	System/36 machine description	QGPL
*TBL	Table	*CURLIB
*USRIDX	User index	*CURLIB
*USRPRF	User profile	N/A
*USRQ	User queue	*CURLIB
*USRSPC	User space	*CURLIB
*WSCST	Work station customizing object	*CURLIB

Table 81 shows object-related commands and the objects on which they operate. The table shows only commands that contain the OBJTYPE parameter and that operate on most objects.

The table includes three major categories:

- Basic object-related commands, which are generally used for normal intrasystem operation.
- The other object-related commands, which are used either for special I/O operations or for diagnostics and low level activity.
- All object-related commands that operate on the DBCS objects *IGCDCT, *IGCSRT, and *ICGTBL. These lines are shown in bold.

The primary purpose of Table 81 is to provide you with a list of the object types that can be operated on by commands containing the OBJTYPE parameter. The Xs indicate that the object type (in the same row as the X) can be specified as a value for the OBJTYPE parameter of the command (in the same column as the X).

Table 81 also provides you with an index to the object-related commands that operate on most objects. The object-related commands allow you to perform general functions on

most objects without knowing the special commands related to the specific object type. For example, you could use the CRTDUPOBJ command to create a copy of a file or library instead of the specific commands CPYF (Copy File) or CPYLIB (Copy Library). The table can help you determine whether you can use the general object-related commands to perform a function on a specific object.

The following commands also contain the OBJTYPE parameter but are not included in the table because they operate on only a few object types.

- CHKDLO operates on *DOC and *FLR.
- CPROBJ and DCPOBJ operate on *FILE, *MENU, *MODULE, *PGM, *PNLGRP, and *SRVPGM.
- CRTSQLPKG operates on *PGM and *SRVPGM.
- DSPCSPOBJ operates on *CSPMAP, *CSPTBL, and *PGM.
- DSPPGMADP operates on *PGM, *SQLPKG, and *SRVPGM.
- DSPPGMREF operates on *PGM and *SQLPKG.
- RSTCFG operates on *CFGL, *CNNL, *COSD, *CTLD, *DEVD, *LIND, *MODD, and *NWID.
- SAVLICPGM operates on *LNG and *PGM.
- SETOBJACC operates on *FILE and *PGM.
- WRKOBJCSP operates on *ALLCSP, *CSPMAP, *CSPTBL, and *PGM.

The ALCOBJ and DLCOBJ commands also require that an object type value is specified. However, for these commands, the object type value is specified as one of four values (in a list of values) on the required parameter OBJ. The following object types can be specified for the OBJ parameter on the ALCOBJ and DLCOBJ commands: *AUTL, *BNDDIR, *CLD, *CSI, *CSPMAP, *CSPTBL, *DEVD, *DTAARA, *DTADCT, *DTAQ, *FCT, *FILE, *FNTRSC, *FORMDF, *LIB, *MENU, *MODULE, *MSGQ, *NODL, *OVL, *PAGDFN, *PAGSEG, *PDG, *PGM, *PNLGRP, *QMFORM, *QMQRY, *QRYDFN, *SBSD, *SCHIDX, *SQLPKG, *SRVPGM, *SSND, *S36, *USRIDX, *USRQ, *USRSPC, and *WSCST.

Note: RTVOBJD and DSPOBJD support the same object types. To see which object types are supported by RTVOBJD, refer to the column labeled DSPOBJD in the following table.

Table 81 (Page 1 of 2). Object Types Used by Commands Containing the OBJTYPE Parameter

	Basic Object-Related Commands																			
	Object					Object Authority					General					Other Object-Related Commands				
Value	CHKOBJ	DMPOBJ	MOVOBJ	RNMOBJ	WRKOBJ	CHGOBJAUD	DSPOBJAUT	EDTOBJAUT	GRTOBJAUT	RVKOBJAUT	CHGOBJD	DSPOBJD	CHGOBJOWN	CRTDUPOBJ	WRKOBJLCK	DMPSYSOBJ	PRTDSKINF	RSTOBJ	SAVCHGOBJ	SAVOBJ
*ALRTBL	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*AUTL	x	x		x	x	x	x	x			x	x	x	x	x	x	x			
*BNDDIR	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CFGL	x	x			x	x	x	x	x	x	x	x	x		x	x	x	x		
*CHTFMT	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CLD	x	x	x		x	x	x	x	x	x			x	x	x	x	x	x	x	x
*CLS	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CMD	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CNNL	x	x		x	x	x	x	x	x	x	x	x	x			x	x			
*COSD	x	x			x	x	x	x	x	x	x	x	x		x	x	x			
*CSI	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CSPMAP	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CSPTBL	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*CTLD	x	x		x	x	x	x	x	x	x	x	x	x		x	x	x			
*DEVD	x	x		x	x	x	x	x	x	x	x	x	x		x	x	x			
*DOC	x	x			x		x					x			x	x	x			
*DTAARA	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*DTADCT	x	x			x	x	x	x	x	x	x	x	x		x	x	x			
*DTAQ	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x	x
*EDTD	x	x		x	x	x	x	x	x	x	x	x	x		x	x	x	x	x	x
*FCT	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*FILE	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*FLR	x	x			x		x					x			x	x	x			
*FNTRSC	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*FORMDF	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*FTR	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*GSS	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*IGCDCT	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x
*IGCSRT	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x		x	x	x
*IGCTBL	x	x			x	x	x	x	x	x	x	x				x		x	x	x
*JOBD	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*JOBQ	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*JOBSCD	x	x			x	x	x	x	x	x	x	x			x	x	x	x	x	x

Table 81 (Page 2 of 2). Object Types Used by Commands Containing the OBJTYPE Parameter

	Basic Object-Related Commands																				
	Object					Object Authority					General					Other Object-Related Commands				
Value	CHKOBJ	DMPOBJ	MOVOBJ	RNMOBJ	WRKOBJ	CHGOBJAUD	DSPOBJAUT	EDTOBJAUT	GRTOBJAUT	RVKOBJAUT	CHGOBJD	DSPOBJD	CHGOBJOWN	CRTDUPOBJ	WRKOBJLCK	DMPSYSOBJ	PRTDSKINF	RSTOBJ	SAVCHGOBJ	SAVOBJ
*JRN	x	x	x		x	x	x	x	x	x	x	x	x		x	x	x	x	x	x
*JRNRCV	x	x	x		x	x	x	x	x	x	x	x	x		x	x	x	x	x	x
*LIB	x	x		x	x	x	x	x	x	x	x	x	x		x	x	x			
*LIND	x	x		x	x	x	x	x	x	x	x	x	x		x	x	x			
*MENU	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*MODD	x	x			x	x	x	x	x	x	x	x	x		x	x	x			
*MODULE	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*MSGF	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*MSGQ	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*NODL	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*NWID	x	x		x	x	x	x	x	x	x	x	x	x		x	x	x			
*OUTQ	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*OVL	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*PAGDFN	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*PAGSEG	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*PDG	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*PGM	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*PNLGRP	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*PRDAVL		x		x		x					x	x	x	x	x	x		x		x
*PRDDFN	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
*PRDLOD		x	x	x	x	x					x	x	x	x	x	x	x	x		
*QMFORM	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*QMQRY	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*QRYDFN	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x
*RCT	x	x	x	x	x	x	x	x	x	x	x	x	x			x	x	x	x	x
*SBSD	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*SCHIDX	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*SPADCT	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x	x
*SQLPKG	x	x			x	x	x	x	x	x	x	x	x		x	x	x	x	x	x
*SRVPGM	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*SSND	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*S36	x	x			x	x	x	x	x	x	x	x	x		x	x	x	x	x	x
*TBL	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*USRIDX	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x
*USRPRF	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x			
*USRQ	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	x	x	x	x
*USRSPC	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
*WSCST	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Values Allowed

**ALL:* All the object types that are allowed in the command, specified by name, and are in the specified library are operated on by the command in which they are specified. *ALL refers only to the object types that apply to that command; refer to the individual command descriptions of the OBJTYPE parameter to see which of the OS/400 object types can be specified.

object-type: Specify the predefined values for the types of objects that are to be operated on by the command.

OUTPUT Parameter

The OUTPUT parameter specifies whether the output from the display command is shown on the display, printed, or written to an output file. Basically, the same information is provided in either form; only the format is changed as necessary to present the information in the best format for the device. For example, because there are more lines on a printed page than on a display, column headings are not repeated as often in printed output.

If the output is shown on the display, it is sent to the work station that issued the display command. It is shown in the format specified in the display device file used by that display command. A different device file is used for the output of each display command, and the file is different for displayed, printed, or written file output. In most cases, the name of the command is part of the file names of either type of device file. See the *Programming Reference Summary* for the names of the spooled printer files and database output files used by each command.

If the output is printed, it is spooled and an entry is placed on the job's output queue. The output can be printed depending on which device is specified in the Start Printer Writer (STRPRTWTR) command.

Note: Although the IBM-supplied printer files are shipped with SPOOL(*YES) specified, they can be changed to SPOOL(*NO) by the Override with Printer File

(OVRPRTF) and Change Printer File (CHGPRTF) commands.

If the OUTPUT parameter is not specified in the display command, the default value * is assumed. The output resulting from this value depends on the type of job that entered the command. The following chart shows how the output is produced for interactive and batch jobs.

Output	Interactive Job	Batch Job
*	Displayed	Printed
*PRINT	Printed	Printed

Values Allowed

***:** Output requested by an interactive job is shown on the display. Output requested by a batch job is printed with the job's spooled output.

**PRINT:* The output is spooled for printing after job completion.

**OUTFILE:* The only output is to be written to a specified database file.

PRTTXT Parameter

The print text (PRTTXT) parameter specifies the text that appears at the bottom of listings and on separator pages. Print text is copied from the job attribute when the job enters the system. Print files that originate on another system do not use the print text on the target system. Print text exists as a job attribute (PRTTXT) for defining the print text of a specific job, and as a system value (QPRTTXT) for the default of jobs with *SYSVAL specified. QPRTTXT is the system-wide default for all jobs.

The print text can be up to 30 characters in length. The text is centered in the form's width and printed in the overflow area. The user should center the desired text within the 30 character field.

If the print text is not blank, the system prints 30 characters of text on the bottom of each page. This text normally follows the overflow line and is preceded by a blank line (if the form's length permits). If the user prints past the overflow line, the print text follows the last line of the user text, again preceded by a blank line when possible. If the overflow line is the last line of the form, the print text also prints on the last line of the form, which may result in the typing over of user text.

The print text for job and file separators is put on the first line of the separator page. A job separator contains print text of the job that created the separator at the time the file was printed. A file separator contains the same print text as the spooled file it precedes.

The print text can be specified for all job types. System and subsystem monitor jobs use the system value. Reader and writer jobs use the system value unless print text is changed in the QSPLxxxx job description associated with the reader or writer.

The print text is determined from several places by using the following hierarchical order. If print text is not specified in one place, the next place in the order is used.

The hierarchical order, beginning with the highest priority, is as follows:

- Override print file value
- Print file value
- Job attribute changed by the Change Job (CHGJOB) command
- Job attribute set by the Submit Job (SBMJOB) or Batch Job (BCHJOB) command
- Job description
- System value

Values Allowed

For the system value QPRTTXT, any character string can be specified, with the exception of *SYSVAL. If *BLANK is specified, there will be no print text. For PRTTXT, some of the following values can be selected, depending on the command:

**SAME:* The print text does not change.

**CURRENT:* The print text is taken from the submitting job.

**JOBD:* The print text is taken from the job description under which the job is run.

**SYSVAL:* The print text is taken from the system value QPRTTXT.

**BLANK:* There is no text or blanks printed.

'print-text': Specify 30 characters of text. If there are blanks in the text, then apostrophes must be used around the entry. The text should be centered within the field for the text to be centered on the page.

REPLACE Parameter

The replace (REPLACE) parameter is used on create commands. It specifies that the existing object, if one exists, is replaced by the object of the same name, library, and object type that is being created. The user of the new object is granted the same authority as for the object being replaced. If the object being replaced is secured by an authorization list, then the new object is secured by the same authorization list. The public authority of the new object is the same as the public authority of the replaced object. The AUT parameter from the create command is ignored. All private authorities from the replaced object are copied to the new object.

The owner of the new object is *not* copied from the replaced object. The owner of the new object is the creator of the new object or the creator's group profile.

If the object being created is a program, then the user profile (USRPRF parameter) value from the replaced program is used. The user profile (USRPRF parameter) value from the Create Program command is ignored. If the value of the user profile (USRPRF parameter) of the program being replaced is *OWNER, then only the current owner of the program being replaced can create the new program that replaces the existing program.

If the object being created is a file, and the default, or *YES, is specified on the REPLACE parameter, an existing device file other than save file and a DDM file with the same qualified name will be replaced by the new file. For example, an existing display file can be replaced by a new printer file, or tape file, etc.

Object management (*OBJMGT), object existence (*OBJEXIST), and read (*READ) authorities are required for the existing object to allow replacement of the existing object with a new object.

The existing object is renamed and moved to library QRPLOBJ when the creation of the new object is successful. The replaced object is renamed with a Q appended to a time stamp and moved to library QRPLOBJ. The text of the replaced object is changed to the original name of the object that was replaced, for example, LIBRARY/OBJNAME.

Restriction: Programs can be replaced while they are being run; however, if the replaced program refers to the program message queue after the renaming of the replaced program to the Qtimestamp name, the program fails and an error message is sent stating that the program message queue is not found.

A database file, physical or logical, and a save file cannot be replaced by any file.

Library QRPLOBJ is cleared when an initial program load (IPL) of the system is done.

Values Allowed

**YES:* The system replaces the existing object with the new object being created that has the same name, library, and object type.

**NO:* The system does not replace the existing object that has the same name, library, and object type with the object being created.

Scheduling Priority Parameters (JOBPTY, OUTPTY, PTYLMT)

The scheduling priority parameters specify the priority values used by the system to determine the order in which the jobs and spooled files are selected for processing. Each job is given a scheduling priority that is used for both job selection and spooled file output. The job scheduling priority is specified by the JOBPTY parameter in commands like the Batch Job (BCHJOB), Submit Job (SBMJOB), Create Job Description (CRTJOBD), and Change Job Description (CHGJOBD) commands. The priority for producing the spooled output from a job is specified by the OUTPTY parameter in the same commands.

In addition, because every job is processed under a specific user profile, the priority for jobs can be limited by the PTYLMT parameter specified on the Create User Profile (CRTUSRPRF) and Change User Profile (CHGUSRPRF) commands. This parameter value controls the maximum job scheduling priority and output priority that *any* job running under a user profile can have; that is, the priority specified in the JOBPTY and OUTPTY parameters of any job command cannot exceed the priority specified in the PTYLMT parameter for that user profile. The scheduling priority is used to determine the order in which jobs are selected for processing and is not related to the process priority specified in the class object.

The three scheduling priority parameters specify the following:

- The PTYLMT parameter specifies the *highest* scheduling priority for *any* job that the user submits. In the commands that affect the user's user profile, the PTYLMT parameter specifies the highest priority that can be specified in another JOBPTY parameter on commands relating to each specific job. The user can specify a lower priority for a job on the command used to submit the job. If a higher priority is specified for JOBPTY in the BCHJOB or SBMJOB command than is specified for PTYLMT in the associated user profile, an error message is shown on the display and the maximum priority specified in PTYLMT is assumed. If a higher job priority is specified in the CHGJOB or CHGJOBD command, an error message is shown and the attributes are not changed.
- The JOBPTY parameter specifies the priority value to be used for a *specific* job being submitted. In the commands relating to a specific job being submitted, the JOBPTY parameter specifies the actual scheduling priority for the job.
- The OUTPTY parameter specifies the priority for producing the output from all spooled output files from the job. The priority value specified in the OUTPTY parameter determines the order in which spooled files are handled for output. The same value is applied to all the spooled files produced by the job.

The scheduling priority can have a value ranging from 0 through 9, where 1 is the highest priority and 9 is the lowest priority. Any job with a priority of 0 is scheduled for processing before all other jobs that are waiting and that have priorities of 1 through 9.

The priority parameters can be specified on the following commands.

JOBPTY	OUTPTY	PTYLMT
BCHJOB	BCHJOB	CHGUSRPR
CHGJOB	CHGDKTF	CRTUSRPR
CRTJOBD	CHGJOB	RTVUSRPR
SBMJOB	CHGJOBD	
	CHGPJ	
	CHGPRTF	
	CHGSPLFA	
	CRTDKTF	
	CRTJOBD	
	CRTPRTF	
	OVRDKTF	
	OVRPRTF	
	SBMJOB	

Values Allowed

Depending upon the command, one or more of the following values apply to the parameter.

5: If a value is not specified in the CRTUSRPRF command, five is the default value that is assumed for the priority limit for the user profile. That would be the highest priority that the user could specify for any job he submits for processing. If not specified in the CRTJOBD command, five is the default value for both the job scheduling priority and the output priority.

**SAME:* The priority assigned, or the highest priority that can be assigned, does not change.

**JOBD:* The scheduling priority for the job is obtained from the job description under which the job runs.

scheduling-priority: Specify a priority value ranging from 0 through 9, where 0 is the highest priority and 9 is the lowest priority. Priority 0 is allowed only on CHGJOB.

SEV Parameter

The severity (SEV) parameter specifies the severity code that:

- Describes the level of severity associated with an error message.
- Indicates the minimum severity level that causes a message to be returned to a user or program.
- Causes a batch job to end.
- Causes processing of a command to end if a syntax error of sufficient severity occurs.

Note: The LOG parameter on some commands also uses these severity codes for logging purposes (to control which job activity messages and error messages are logged in the job log).

The severity code is a 2-digit number that can range from 00 through 99. The higher the value, the more severe or important the condition. The severity code of a message that is sent to a user indicates the severity of the condition described by the message. More than one message can have the same severity code. If a severity code is not specified for a predefined message, it is assumed to be 00 (information only).

The user can specify a severity code for any message when it is defined by the Add Message Description (ADDMSGD) command. To change the severity code of a message, use the Change Message Description (CHGMSGD) command.

IBM-defined severity codes are used in all of the IBM-supplied messages that are shipped with the system.

00 – Information: A message of this severity is for information purposes only; no error was detected and no reply is needed. The message could indicate that a function is in progress or that it has reached a successful completion.

10 – Warning: A message of this severity indicates a potential error condition. The program may have taken a default, such as supplying missing input. The results of the operation are assumed to be what was intended.

20 – Error: An error has been detected, but it is one for which automatic recovery procedures probably were applied, and processing has continued. A default may have been taken to replace input that was in error. The results of the operation may not be valid. The function may be only partially complete; for example, some items in a list may be processed correctly while others may fail.

30 – Severe Error: The error detected is too severe for automatic recovery, and no defaults are possible. If the error was in source data, the entire input record was skipped. If the error occurred during program processing, it leads to an abnormal end of the program (severity 40). The results of the operation are not valid.

40 – Abnormal End of Program or Function: The operation has ended, possibly because it was unable to handle invalid data, or possibly because the user ended it.

50 – Abnormal End of Job: The job was ended or was not started. A routing step may have ended abnormally or failed to start, a job-level function may not have been performed as required, or the job may have been ended.

60 – System Status: A message of this severity is issued only to the system operator. It gives either the status of or a warning about a device, a subsystem, or the whole system.

70 – Device Integrity: A message of this severity is issued only to the system operator. It indicates that a device is malfunctioning or in some way is no longer operational. The

user may be able to restore system operation, or the assistance of a service representative may be required.

80 – System Alert: A message of this severity is issued only to the system operator. It warns of a condition that, although not severe enough to stop the system now, could become more severe unless preventive measures are taken.

90 – System Integrity: A message of this severity is issued only to the system operator. It describes a condition that renders either a subsystem or the whole system inoperative.

99 – Action: A message of this severity indicates that some manual action is required, such as specifying a reply, changing printer forms, or replacing diskettes.

SPLNBR Parameter

The spooled file number (SPLNBR) parameter is used when more than one spooled file is created by a job and the files all have the same name. The files are numbered, starting with 1, in the order that they are opened by the job. The job log is always the last file for a job.

A file number is generated for each file when it is opened within a job (when output records are produced) and it is used by the system as long as the job and/or the files are on the system. If the files are not uniquely named because they were opened more than once, this file number is used to specify which file (or group of records, if the complete file has not yet been produced) is acted upon by a CL command.

TEXT Parameter

The TEXT parameter specifies the user-defined description that briefly describes the object being created or changed. The description can include up to 50 characters; if it is a quoted string (that is, enclosed in apostrophes), any of the 256 EBCDIC characters can be used. The apostrophes are not required if the string does not contain any blanks or other special characters. Any of the 50 character positions not filled by the specified description are padded with blanks.

The description is used to describe any of the OS/400 objects when the named object is shown on the display by the Display Object Description (DSPOBJD) command. Only objects for which object operational authority has been obtained can be displayed by a user. See the OBJTYPE parameter description in this appendix for a list of the OS/400 object types.

For commands that use a database source file to create some type of object, you can (by default) use the text from the source file member as the text for the newly-created object. For example, if you use the Create Control Language Program (CRTCLPGM) command to create a CL program, but you do not specify a description in the TEXT parameter, the text specified for the source file member (SRCMBR parameter) of the source file (SRCFILE parameter) is assumed as the descriptive text for the CL program.

Values Allowed

Depending upon the command, one or more of the following values apply to the TEXT parameter.

**SRCMBRTXT:* For commands that create objects based on database source files only, the text is taken from the source member. If a device or an inline file is used for source input or if source is not used, the text is left blank.

**BLANK:* The user description of the object being created or changed is left blank.

**SAME:* The user-defined description does not change.

'description': Specify the description of the object being created or changed. Up to 50 characters enclosed in apostrophes (required for blanks and other special characters) can be specified to describe the object. If an apostrophe is one of the 50 characters, two apostrophes (") must be used instead of one to represent the apostrophe character.

VOL Parameter

The volume (VOL) parameter specifies the volume identifiers of the volumes used in a diskette or tape operation. A volume consists of a single diskette or reel of tape; each diskette or reel is a separate volume.

The volume identifier is the identifier stored on each diskette or tape (in the volume label area) that it identifies. The diskettes (volumes) must be on the diskette drive in the same order as the identifiers are specified in the VOL parameter. An inquiry message is sent to the system operator if a volume identifier is missing or out of order.

Tape volumes must be on the tape units in the same order as their identifiers are specified in the VOL parameter and as the device names are specified in the DEV parameter of the tape device file commands. However, if the tapes are read backward (a function supported in COBOL), the volumes must be in reverse order to that specified in the VOL parameter. Nevertheless, the device names are still specified in forward order in the DEV parameter.

The general rule for specifying diskette and tape volume identifiers is that as many as 6 characters, containing any combination of letters and digits, can be used. Special characters can be used if the identifier is enclosed in apostrophes. However, if the diskette or tape is used on a system other than an AS/400 system, the requirements for specifying identifiers on that system must be considered.

For diskettes in the data exchange format and for labeled tapes, the following rules apply:

- Characters: A maximum of 6 characters, or fewer, can be specified for each volume identifier. Alphabetic and numeric characters can be used in any order.

- Uniqueness: More than one volume can have the same identifier. You may have a file using the same identifier for several volumes; in this case, the system keeps track of the order internally with a sequence number written on the volumes. However, volume identifiers should be unique whenever possible.
- Order: When multiple volumes (with different identifiers) are used in a single operation, they must be in the same order as the volume identifiers specified in the VOL parameter.

Multivolume Files

For a multivolume file on diskettes (that is, a data file on several diskettes, all having the same name), a message is sent to the system operator for each diskette volume after the first, until all volumes have been processed. If (for S/R only) more than 100 diskettes are used for the same file, duplicate diskette sequence numbers occur for each additional diskette used after the first hundred (01 through 99, and 00). For each 100 diskettes written for the file, a message is sent to the system operator indicating the total number written. When the diskettes are read, the operator must determine the order in which the diskette volumes are inserted.

If multiple volumes (tapes or diskettes) are used in an operation and all have the same volume identifier, that identifier must be specified in the VOL parameter once for each volume used. For example, if three tapes named QGPL are used in a save operation, VOL(QGPL QGPL QGPL) must be specified.

When a multivolume file on *tape* is processed and multiple tape units are used, the tape volumes must be placed in the tape devices in the same order as they are specified in the VOL parameter. For example, if five volumes and three tape units are used, they are mounted as follows: VOL1 on unit 1, VOL2 on unit 2, VOL3 on unit 3, VOL4 on unit 1, and VOL5 on unit 2.

Values Allowed

**MOUNTED:* The volumes mounted in the unit are used for the operation.

**NONE:* No volume identifier is specified.

**SAME:* Previously specified volume identification does not change.

**SAVVOL:* The system, using the save/restore history information, determines which tape or diskette volumes contain the most recently saved version. If the device specified in the DEV parameter of the restore command does not match the device of the most recently saved version of the object, an error message is returned to the user, and the function is ended. If the wrong volume is mounted in the unit specified by the command, a message is returned to the system operator that identifies the first volume that must be placed in the device before the restore operation can begin.

volume-identifier: Specify the identifiers of one or more volumes in the order in which they must be placed on the device and used. Each volume identifier contains a maximum of six characters.

WAITFILE Parameter

The WAITFILE parameter specifies the maximum number of seconds that a program waits for file resources to be allocated when the file is opened, for session resources when the evoke function is issued for an APPC device, and for the device to be allocated when an acquire operation is performed to read the file. If the program must wait, it is placed in a wait state until the resources are available or until the wait time expires. If two or more file resources are needed and are not available because they are being used by different system users, the acquisition of each resource might require a wait. This maximum is applied to each wait.

The length of the wait can be specified in this parameter, or the default wait time of the class that applies to the object can be used. If the file resources cannot be allocated in the specified number of seconds, an error message is returned to the program.

The file resources that must be allocated depend on the type of file being opened. File resources consist of the following.

- For device files that are not spooled (SPOOL(*NO)), the file resources include the file description and device description. Because the device description must be allocated, the device itself must also be available.
- For device files that are spooled (SPOOL(*YES)), the file resources include the file description, the specified output queue, and storage in the system for the spooled data. Because the data is spooled, the device description (and thus the device itself) need not be available.
- For database files, the file resources consist of the file and member data. The file's associated member paths are not accessed, and therefore, the system does not wait for them. A file open exception error can occur before the WAITFILE time has expired when an access path is not available (for example, when the access path is being rebuilt).

  The Allocate Object (ALCOBJ) command can be used to allocate specific file resources before the file is opened.

The session resources that were allocated for an APPC device conversation can be lost between the time the application issues a detach function or receives a detach indication and the time another evoke function is issued. If the session resource is lost, this parameter is used to determine the length of time that the system waits for another session resource.

Values Allowed

**IMMED:* This value specifies that no wait time is allowed. When the file is opened, an immediate allocation of the file resources is required.

**CLS:* The default wait time specified in the class description is used as the wait time for the file resources to be allocated.

number-of-seconds: Specify the maximum number of seconds that the program waits for the file resources to be allocated. Valid values range from 1 through 32767 seconds.

Appendix B. Font, Character Identifier, and Other Values Supported for Different Printers

The tables in this appendix contain information on fonts, character identifiers and other printing characteristics.

Font Attributes

Font attributes are the characteristics or properties that combine to give a font identity. For example: attributes can be 14 point (height of the font), bold, and italic.

Types of Fonts

The following diagram identifies the types of fonts and gives examples of each type:

- Mixed pitch fonts which simulate proportionally spaced fonts.

  Characters in the font have a limited number of widths. Overall spacing is about 12 characters per inch. Examples are Document or Essay fonts.

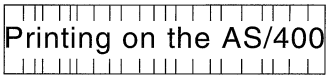
- Uniformly spaced fonts which are similar to typewriter fonts.

  Characters in the font are all the same width. Examples are Courier and Gothic Text fonts.

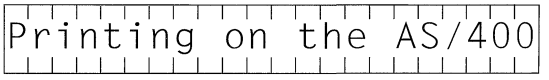
- Typographic fonts

  Typographic fonts have variable height, measured in points (1 point = 1/72 inch). Therefore, a 36-point font has characters that are 1/2 inch high. Typographic fonts have variable widths. Width is part of the design and varies on a character-by-character basis. Examples are Sonoran Serif and Century Schoolbook.

Mixed Pitch



Uniformly Spaced



Typographic

Printing on the AS/400	6 pt Century Schoolbook
Printing on the AS/400	8 pt Century Schoolbook
Printing on the AS/400	10 pt Century Schoolbook

RV2H301-2

Font, Character Identifier, and Other Values Supported for Different Printers

The legend and table below provide information about each font. This information could save you time in trial-and-error testing when choosing a font for your application.

Table Legend	
FGID	Font Global Identifier
Name	Name of Font
Font Type	**U** = Uniformly Spaced **M** = Mixed Pitch **T** = Typographic
Attributes	**Blank** = Roman **b** = Bold **i** = Italics **s** = Second Strike **w** = Double Wide
Point	Point size (Blank for uniformly spaced & mixed pitch fonts)
Pitch	Characters per inch

Table 82 (Page 1 of 6). Font Information

FGID	Name	Type of Font	Attri-butes	Point	Pitch (CPI)
2	Delegate	U			10
3	OCR-B	U			10
5	Rhetoric/Orator	U			
8	Scribe/Symbol	U			10
10	Cyrillic 22	U			10
11	Courier	U			10
12	Prestige	U			10
13	Artisan	U			10
18	Courier Italic	U	I		10
19	OCR-A	U			10
20	Pica	U			10
21	Katakana	U			10
25	Presentor	U			10
26	Matrix Gothic	U			10
30	Symbol	U			10
31	Aviv	U			10
36	Letter Gothic	U			10
38	Orator Bold	U	b		10
39	Gothic Bold	Ub	b		10
40	Gothic	U			10
41	Roman Text	U			10
42	Serif	U			10
43	Serif Italic	U	i		10
44	Katakana Gothic	U			10
46	Courier Bold	U	b		10
49	Shalom	U			10
50	Shalom Bold	U	b		10
51	Matrix Gothic	U			10
52	Courier	U			10
55	Aviv Bold	U	b		10
61	Nasseem	U			10
62	Nasseem Italic	U	i		10
63	Nasseem Bold	U	b		10
64	Nasseem Italic Bold	U	bi		10
66	Gothic	U		12	
68	Gothic Italic	U	i		12
69	Gothic Bold	U	b		12
70	Serif	U			12
71	Serif Italic	U	i		12
72	Serif Bold	U	b		12
74	Matrix Gothic	U			12

Table 82 (Page 2 of 6). Font Information

FGID	Name	Type of Font	Attri-butes	Point	Pitch (CPI)
75	Courier	U			12
76	APL	U			12
78	Katakana	U			12
80	Symbol	U			12
84	Script	U			12
85	Courier	U			12
86	Prestige	U			12
87	Letter Gothic	U			12
91	Light Italic	Ui			12
92	Courier Italic	U	i		12
95	Adjudant	U			12
96	Old World	U			12
98	Shalom	U			12
99	Aviv	U			12
101	Shalom Bold	U	b		12
102	Aviv Bold	U	b		12
103	Nasseem	U			12
109	Letter Gothic Italic	U	i		12
110	Letter Gothic Bold	U	b		12
111	Prestige Bold	U	b		12
112	Prestige Italic	U	i		12
154	Essay	M			12
155	Boldface Italic	M	bi		12
157	Title	M			12
158	Modern	M			12
159	Boldface	M	b		12
160	Essay	M			12
162	Essay Italic	M	i		12
163	Essay Bold	M	b		12
164	Prestige	M			12
167	Barak	M			12
168	Barak Bold	M	b		12
173	Essay	M			12
174	Gothic	M			12
175	Document	M			12
178	Barak	M			18
179	Barak Bold	M	b		18
180	Barak	M			15
181	Barak (8) Mixed Bold	M	b		15
182	Barak	M			5
183	Barak Bold	M	b		5
186	Press Roman	M			12
187	Press Roman Bold	M	b		12
188	Press Roman Italic	M	i		12
189	Press Roman Italic Bold	M	bi		12
190	Foundry	M			12
191	Foundry Bold	M	b		12
194	Foundry Italic	M	i		12
195	Foundry Italic Bold	M	bi		12

Table 82 (Page 3 of 6). Font Information

FGID	Name	Type of Font	Attri-butes	Point	Pitch (CPI)
204	Matrix Gothic	U			13
205	Matrix Gothic	U			13
211	Shalom	U			15
212	Shalom Bold	U	b		15
221	Prestige	U			15
222	Gothic	U			15
223	Courier	U			15
225	Symbol	U			15
226	Shalom	U			15
229	Serif	U			15
230	Gothic	U			15
232	Matrix Gothic	U			15
233	Matrix Courier	U			15
234	Shalom Bold	U	b		15
244	Courier Double Wide	U	w		5
245	Courier Bold Double Wide	U	wb		5
247	Shalom Bold	U	b		17
248	Shalom	U			17
249	Katakana	U			17
252	Courier	U			17
253	Courier Bold	U	b		17
254	Courier	U			17
255	Matrix Gothic	U			17
256	Prestige	U			17
258	Matrix Gothic	U			18
259	Matrix Gothic	U			18
279	Nasseem	U			17
281	Gothic Text	U			20
282	Aviv	U			20
283	Letter Gothic	U			20
285	Letter Gothic	U			25
290	Gothic Text	U			27
300	Gothic	U			17
400	Gothic	U			17
434	Orator Bold	U	b		8
435	Orator Bold	U	b		6
751	Sonoran Serif	T		8P.	27
752	Nasseem	T		12P	18
753	Nasseem Bold	T	b	12P	18
754	Nasseem Bold	T	b	18P	12
755	Nasseem Bold	T	b	24P	9
756	Nasseem Italic	T	i	12P	18
757	Nasseem Bold Italic	T	bi	12P	18
758	Nasseem Bold Italic	T	bi	18P	12
759	Nasseem Bold Italic	T	bi	24P	9
760	Times Roman	T		6P	36
761	Times Roman Bold	T	b	12P	18
762	Times Roman Bold	T	b	10P	15
763	Times Roman Italic	T	i	12P	18

Table 82 (Page 4 of 6). Font Information

FGID	Name	Type of Font	Attri-butes	Point	Pitch (CPI)
764	Times Roman Bold Italic	T	bi	10P	21
765	Times Roman Bold Italic	T	bi	12P	18
1051	Sonoran Serif	T		10P	21
1053	Sonoran Serif Bold	T	b	10P	21
1056	Sonoran Serif Italic	T	i	10P	21
1351	Sonoran Serif	T		12P	18
1653	Sonoran Serif Bold	T	b		13
1803	Sonoran Serif Bold	T	b	18P	12
2103	Sonoran Serif Bold	T	b	24P	9
4407	Sonoran Serif	T		8P	*27
4407	Sonoran Serif	T		10P	*21
4407	Sonoran Serif	T		12P	*18
4427	Sonoran Serif Bold	T	b	10P	*21
4427	Sonoran Serif Bold	T	b	16P	*13
4427	Sonoran Serif Bold	T	b	24P	*9
4535	Sonoran Serif Italic	T	i	10P	*21
4919	Goudy	T		6P	*36
4919	Goudy	T		8P	*27
4919	Goudy	T		10P	*21
4919	Goudy	T		12P	*18
4939	Goudy Bold		Tb	10P	*21
4939	Goudy Bold	T	b	14P	*15
4939	Goudy Bold	T	b	18P	*12
5047	Goudy Italic	T	i	10P	*21
5067	Goudy Bold Italic	T	bi	10P	*21
5687	Times Roman	T		6P	*36
5687	Times Roman	T		8P	*27
5687	Times Roman	T		10P	*21
5687	Times Roman	T		12P	*18
5707	Times Roman Bold	T	b	10P	*21
5707	Times Roman Bold	T	b	12P	*18
5707	Times Roman Bold	T	b	14P	*15
5707	Times Roman Bold	T	b	18P	*12
5707	Times Roman Bold	T	b	24P	*12
5815	Times Roman Italic	T	i	10P	*21
5815	Times Roman Italic	T	i	12P	*18
5835	Times Roman Italic Bold	T	bi	10P	*21
5835	Times Roman Italic Bold	T	bi	12P	*18
5943	University	T		12P	*18
5943	University	T		14P	*15
5943	University	T		18P	*12
6199	Palatino**	T		6P	*36
6199	Palatino	T		8P	*27
6199	Palatino	T		10P	*21
6199	Palatino	T		12P	*18
6219	Palatino Bold	T	b	10P	*21
6219	Palatino Bold	T	b	14P	*15
6219	Palatino Bold	T	b	18P	*12
6327	Palatino Italic	T	i	10P	*21

Table 82 (Page 5 of 6). Font Information

FGID	Name	Type of Font	Attri-butes	Point	Pitch (CPI)
6347	Palatino Italic Bold	T	bi	10P	*21
8503	Baskerville	T		6P	*36
8503	Baskerville	T		8P	*27
8503	Baskerville	T		10P	*21
8503	Baskerville	T		12P	*18
8523	Baskerville Bold	T	b	10P	*21
8523	Baskerville Bold	T	b	14P	*15
8523	Baskerville Bold	T	b	18P	*12
8631	Baskerville Italic	T	i	10P	*21
8651	Baskerville Italic Bold	T	bi	10P	*21
8759	Nasseem	T		12P	*18
8779	Nasseem Bold	T	b	12P	*18
8779	Nasseem Bold	T	b	18P	*12
8779	Nasseem Bold	T	b	24P	*9
8887	Nasseem Italic	T	i	12P	*18
8907	Nasseem Italic Bold	T	bi	12P	*18
8907	Nasseem Italic Bold	T	bi	18P	*12
8907	Nasseem Italic Bold	T	bi	24P	*9
12855	Narkisim	T		8P	*27
12855	Narkisim	T		10P	*21
12855	Narkisim	T		18P	*12
12855	Narkisim	T		24P	*9
12875	Narkisim Bold	T	b	8P	*27
12875	Narkisim Bold	T	b	10P	*21
12875	Narkisim Bold	T	b	12P	*18
16951	Century Schoolbook**	T		6P	*36
16951	Century Schoolbook	T		8P	*27
16951	Century Schoolbook	T		10P	*21
16951	Century Schoolbook	T		12P	*18
16971	Century Schoolbook Bold	T	b	10P	*21
16971	Century Schoolbook Bold	T	b	14P	*15
16971	Century Schoolbook Bold	T	b	18P	*12
17079	Century Schoolbook Italic	T	i	10P	*21
17099	Century Schoolbook Italic Bold	T	bi	10P	*21
33335	Optima**	T		6P	*36
33335	Optima	T		8P	*27
33335	Optima	T		10P	*21
33335	Optima	T		12P	*18
33355	Optima Bold	T	b	10P	*21
33355	Optima Bold	T	b	14P	*15
33355	Optima Bold	T	b	18P	*12
33463	Optima Italic	T	i	10P	*21
33483	Optima Italic Bold	T	bi	10P	*21
33591	Futura**	T		6P	*36
33591	Futura	T		8P	*27
33591	Futura	T		10P	*21
33591	Futura	T		12P	*18
33601	Futura Bold	T	b	10P	*21
33601	Futura Bold	T	b	14P	*15

Table 82 (Page 6 of 6). Font Information

FGID	Name	Type of Font	Attri-butes	Point	Pitch (CPI)
33601	Futura Bold	T	b	18P	*12
33719	Futura Italic	T	i	10P	*21
33729	Futura Italic Bold	T	bi	10P	*21
34103	Helvetica**	T		6P	*36
34103	Helvetica	T		8P	*27
34103	Helvetica	T		10P	*21
34103	Helvetica	T		12P	*18
34123	Helvetica Bold	T	b	10P	*21
34123	Helvetica Bold	T	b	14P	*15
34123	Helvetica Bold	T	b	18P	*12
34231	Helvetica Italic	T	i	10P	*21
34251	Helvetica Italic Bold	T	bi	10P	*21
37431	Old English	T		12P	*18
37431	Old English	T		14P	*15
37431	Old English	T		18P	*12
41783	Coronet Cursive	T		12P	*18
41803	Coronet Cursive Bold	T	b	14P	*15
41803	Coronet Cursive Bold	T	b	18P	*12

Note: Pitch or CPI column for typographic fonts indicates the width of the space character between printed characters. Width, pitch, and CPI of other space characters will vary.

Font Substitution

Font substitution is done by the AS/400 system when the application specifies a font ID that is not supported by the designated printer or cannot be downloaded from the AS/400 system to the designated printer.

Table 83 lists many fonts (by FGID number) and printers that are supported. A blank in any column indicates that the font ID is supported by that printer, and no substitution takes place. However, if your application specifies a font ID that is not in the table, you need to refer to Table 75. Table 75 provides the substituted FGID for font IDs in ranges such as FGID 0 through FGID 65.

How To Use the Font Substitution Charts

Following are three examples to familiarize you with font substitution on the AS/400 system.

- Example one shows how to verify whether or not your font ID is supported by a certain printer.
- Example two shows how to find out what font ID the AS/400 system substitutes if the printer you want to use does not support your font ID.
- Example three shows how to find out what font ID the AS/400 system substitutes if your font ID is not available on the AS/400 system or on the printer.

Example One: If you want to verify that a font ID is supported by a certain printer, locate the font ID in Table 83. For example, locate font ID 112. Font ID 112 is supported by the 3812 and 3816 SCS and IPDS printers and the 4028 printer (this is indicated by blanks in those spaces). The 4019 printer supports font ID 112 on a font card resident in the 4019 printer. The 4224, 4234, and 5219 printers substitute font ID 87 or 86.

Note: A font card is a hardware card that can have many font character sets resident on it. Font cards can be installed in printers to provide additional fonts.

Example Two: If your application uses a font ID that is not supported on all printers, you can determine the substitution by looking in Table 83. For example, locate font ID 30. The table shows that font ID 30 is supported on the 3812 and 3816 SCS and IPDS printers. However, if you are using any of the other printers listed in the table, font ID 11 is substituted for font ID 30.

Example Three: Let us say your application calls for font ID 4 and you want to print the spooled file on a 4224 printer. To determine if font ID 4 is a supported font or one that is substituted for, read through the following steps:

Step 1 Look in Table 83 to see if font ID 4 is listed. Font ID 4 is not in Table 83.

Step 2 Next, look in Table 75. The table shows that font ID 11 is substituted for fonts 0 through 65.

Step 3 Return to Table 83 and locate font ID 11. This table shows that font ID 11 is supported on the 4224 printer.

Step 4 The result of the font ID substitution is that your application will print using font ID 11.

Changing Font IDs: To permanently change the font ID, you could, in your application, specify a different font ID or use the Change Printer File (CHGPRTF) command to specify a new font ID for the printer file. Information in Table 82 can help you choose a replacement font ID.

To temporarily change the font ID for your application, you could override the font selection in your printer file by using the Override with Printer File (OVRPRTF) command before the application runs.

Font Substitution and the 4019 Printer

The 4019 printer is supported by the AS/400 system, as an emulated printer (usually 3812 or 5219). The AS/400 system treats the device as a physical 3812 or 5219. Therefore, the font support and font substitution of the emulated printer is used. This emulation limits access to some of the 4019 fonts.

To access most of the 4019-supported fonts, an IBM-supplied program named QWP4019 is available. QWP4019 sets a flag in the emulated printer's device description to inform the system to use the 4019 font tables.

Note to Reader

An asterisk is used in the following chart to indicate that the substituted font has a different pitch.

Table 83 (Page 1 of 6). Font Substitution

	Printers						
FGID	4224 4230	4234	3812 or 3816 SCS	3812 or 3816 IPDS	5219	4028	4019[1]
2	11	11	11	11	11	11[2]	
3					11		
5	11	26				11[2]	
8	11	11	11	11	11	11	
10	11	11	11	11	11	11	
11							
12	11	26					
13	11	11				11	11
18	11	26			11		
19					11		
20	11	26				11	11
21	11	11	11	11	11	11	
25	11	11	11	11	11	11[2]	
26						11	11
30	11	11			11	11	11
31	26	26	26		26	11	11
36	11	11	11	11	11	11[2]	
38	11	26			11	46	46
39	26	26			11	46	46
40	26	26			11	11	11
41	11	26			11	11	11
42	11	26			11	11	11
43	11	26			11	18	11
44	11	11			11	11	11
46	11	26			11		
49	26	26	26		26	11	
50	26	26			26	46	
51	26				26	11	11
52	11				11	11	11
55	26	26	26		26	46	46
61	11	11	11	11	11	11	

Table 83 (Page 2 of 6). Font Substitution

FGID	Printers						
	4224 4230	4234	3812 or 3816 SCS	3812 or 3816 IPDS	5219	4028	4019[1]
62	11	11	11	11	11	18	
63	11	11	11	11	11	46	
64	11	11	11	11	11	46	
66	87	87			87	85	85
68	87	87			87	92	85
69	87	87			87	111	85
70	87	87			87	85	85
71	87	87			87	92	85
72	87	87			87	111	85
74	87		87	87	87	85	85
75	85		85	85	85	85	85
76	85	85	85	85	85		
78	85	85	85	85	85	85	
80	87	87				85	
84	87	87				85[2]	
85							
86	87	87					
87						85[2]	
91	87	87				92[2]	
92	85	85	85	85	85		
95	85	85	85	85	85	85[2]	
96	85	85	85	85	85	85[2]	
98	87	87	87		87	85	
99	87	87	87		87	85	85
101	87	87	87		87	111	85
102	87	87	87		87	111	85
103	85	85	85	85	85	85	
109	85	85	85	85	85	92[2]	
110	87	87			87	11[2]	
111	87	87			86		
112	87	87			86		
154	85		160	160	160	164	159
155	160	160			160	159[2]	
157	160	160	160	160	160	164[2]	
158	160	160				164[2]	
159	160	160					
160						164[2]	
162	160	160				164[2]	
163	160	160			160	159	159
164	160	160	160	160	160		
167	160	160	160		160	164	
168	160	160	160		160	159	159
173	160	160			160	164	159
174	160	160	160	160	160	164	159
175	160	160			160	164	159
178	*400	*258	*281		*222	*281	*254
179	*400	*258	*281		*222	*281	*254
180	*222	*222	*230		*222	*223	*254

Table 83 (Page 3 of 6). Font Substitution

FGID	Printers						
	4224 4230	4234	3812 or 3816 SCS	3812 or 3816 IPDS	5219	4028	4019[1]
181	*222	*222	*230		*222	*223	*254
182	*11	*11	*244		*11	*11	*11
183	*11	*11	*244		*11	*46	*46
186	160	160	160	160	160	164[2]	
187	160	160	160	160	160	159[2]	
188	160	160	160	160	160	164[2]	
189	160	160	160	160	160	159[2]	
190	160	160	160	160	160	164[2]	
191	160	160	160	160	160	159[2]	
194	160	160	160	160	160	164[2]	
195	160	160	160	160	160	159[2]	
204	*222				*222	*223	*254
205	*222		204	204	*222	*223	*254
211	222	222	230		222	223	*254
212	222	222	230		222	223	*254
221	222	222					
222			230	230		223[2]	
223							
225	222	222				223	*254
226	222	222	230		222	223	
229	222	222			222	223	*254
230	222	222			222	223	*254
232	222		230	230	222	223	*254
233	223		230	230	223	223	*254
234	222	222	230		222	223	*254
244	*11	*26			*11	*11	
245	*11	*26			*11	*46	
247	*400	*258	252		*222	254	254
248	*400	*258	252		*222	254	254
249	*400	*258	252	252	*222	254	
252	*400	*258			*222	254	254
253	*400	*258			*222	254	254
254	*400	*258			*222.		
255	*400	*258	252	252	*222	254	
256	*400	*258	252	252	*222.		
258	*400		*281	*281	*222	*281	*254
259	*400		*281	*281	*222	*281	*254
279	*400	*258	252	252	*222	254	
281	*400	*258			*222		
282	*400	*258	281		*222	281	
283	*400	*258	281	281.	*222	281	
285	*400	*258	*290	*290	*222	281[2]	
290	*400	*258			*222	*281	*254
300	400		*252	*252	*222	*254	*254
400			*252	*252	*222	*254	*254
434	*11	*11	*11	*11	*11	46[2]	
435	*11	*11	*11	*11	*11	46[2]	
751	*400	*258			*222.		*254

Table 83 (Page 4 of 6). Font Substitution

FGID	Printers						
	4224 4230	4234	3812 or 3816 SCS	3812 or 3816 IPDS	5219	4028	4019[1]
752	*400	*258	*281	*281	*222		*254
753	*400	*258	*281	*281	*222		*254
754	*85	*85	*85	*85	*85		*85
755	*11	*11	*11	*11	*11		*46
756	*400	*258	*281	*281	*222		*254
757	*400	*258	*281	*281	*222		*254
758	*85	*85	*85	*85	*85		*85
759	*11	*11	*11	*11	*11		*46
760	*400	*258	*290	*290	*222		*254
761	*400	*258	*281	*281	*222		*254
762	*222	*222	*230	*230	*222		*254
763	*400	*258	*281	*281	*222		*254
764	*400	*258	*290	*290	*222		*254
765	*400	*258	*281	*281	*222		*254
1051	*400	*258			*222		*254
1053	*400	*258			*222		*254
1056	*400	*258			*222		*254
1351	*400	*258			*222		*254
1653	*222	*222			*222		*254
1803	*85	*85	*85	*85	*85		*85
2103	*11	*11			*11		*46
4407 (8P)	*400	*258			*222	5687[2]	*254
4407 (10P)	*400	*258			*222	5687[2]	*254
4407 (12P)	*400	*258			*222	5687[2]	*254
4427 (10P)	*400	258			*222	5687[2]	*254
4427 (16P)	*222	*222			*11	*5707	*254
4427 (24P)	*11	*11			*11	5707[2]	*254
4535 (10P)	*400	*258			*222	5687[2]	*46
4919 (6P)	*400	*258	*290	*290	*222	5687[2]	
4919 (8P)	*400	*258	*751	*751	*222	5687[2]	
4919 (10P)	*400	*258	*1051	*1051	*222	5687[2]	
4919 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
4939 (10P)	*400	*258	*1053	*1053	*222	5707[2]	
4939 (14P)	*222	*222	*1351	*1351	*222	5707[2]	
4939 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
5047 (10P)	*400	*258	*1056	*1056	*222	5687[2]	
5067 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
5687 (6P)	*400	*258	*290	*290	*222		
5687 (8P)	*400	*258	*751	*751	*222		
5687 (10P)	*400	*258	*1051	*1051	*222		
5687 (12P)	*400	*258	*1351	*1351	*222		
5707 (10P)	*400	*258	*1053	*1053	*222		
5707 (12P)	*400	*258	*1351	*1351	*222		*254
5707 (14P)	*222	*222	*1351	*1351	*222		
5707 (18P)	*85	*85	*1653	*1653	*85		
5707 (24P)	*11	*11	*2103	*2103	*11		
5815 (10P)	*400	*258	*1056	*1056	*222		
5815 (12P)	*400	*258	*1351	*1351	*222		*254

Table 83 (Page 5 of 6). Font Substitution

FGID	Printers						
	4224 4230	4234	3812 or 3816 SCS	3812 or 3816 IPDS	5219	4028	4019[1]
5835 (10P)	*400	*258	*1053	*1053	*222		
5835 (12P)	*400	*258	*1351	*1351	*222		
5943 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
5943 (14P)	*222	*222	*1351	*1351	*222	5707[2]	
5943 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
6199 (6P)	*400	*258	*290	*290	*222	5687[2]	
6199 (8P)	*400	*258	*751	*751	*222	5687[2]	
6199 (10P)	*400	*258	*1051	*1051	*222	5687[2]	
6199 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
6219 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
6219 (14P)	*222	*222	*1351	*1351	*222	5707[2]	
6219 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
6327 (10P)	*400	*258	*1056	*1056	*222	5687[2]	
6347 (10P)	*400	*258	*1053	*1053	*222	5686[2]	
8503 (6P)	*400	*258	*290	*290	*222	5687[2]	
8503 (8P)	*400	*258	*751	*751	*222	5687[2]	
8503 (10P)	*400	*258	*1051	*1051	*222	5687[2]	
8503 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
8523 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
8523 (14P)	*222	*222	*1351	*1351	*222	5707[2]	
8523 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
8631 (10P)	*400	*258	*1056	*1056	*222	5687[2]	
8651 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
8759 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
8779 (12P)	*400	*258	*1351	*1351	*222	5707[2]	
8779 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
8779 (24P)	*11	*11	*2103	*2103	*11	5707[2]	
8887 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
8907 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
8907 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
8907 (24P)	*11	*11	*2103	*2103	*11	5707[2]	
12855 (8P)	*400	*258	*751		*222	5687[2]	
12855 (10P)	*400	*258	*1051	*1051	*222	5687[2]	
12855 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
12855 (24P)	*11	*11	*2103	*2103	*11	5707[2]	
12875 (8P)	*400	*258	*751		*222	5687[2]	
12875 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
12875 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
16951 (6P)	*400	*258	*290	*290	*222	5687[2]	
16951 (8P)	*400	*258	*751	*751	*222	5687[2]	
16951 (10P)	*400	*258	*1051	*1051	*222	5687[2]	
16951 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
16971 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
16971 (14P)	*222	*222	*1351	*1351	*222	5707[2]	
16971 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
17079 (10P)	*400	*258	*1056	*1056	*222	5687[2]	
17099	*400	*258	*1053	*1053	*222	5687[2]	
33335 (6P)	*400	*258	*290	*290	*222	5687[2]	

Table 83 (Page 6 of 6). Font Substitution

FGID	Printers						
	4224 4230	4234	3812 or 3816 SCS	3812 or 3816 IPDS	5219	4028	4019[1]
33335 (8P)	*400	*258	*751	*751	*222	5687[2]	
33335 (10P)	*400	*258	*1051	*1051	*222	5687[2]	
33335 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
33355 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
33355 (14P)	*222	*222	*1351	*1351	*222	5707[2]	
33355 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
33463 (10P)	*400	*258	*1056	*1056	*222	5687[2]	
33483 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
33591 (6P)	*400	*258	*290	*290	*222	5687[2]	
33591 (8P)	*400	*258	*751	*751	*222	5687[2]	
33591 (10P)	*400	*258	*1051	*1051	*222	5687[2]	
33591 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
33601 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
33601 (14P)	*222	*222	*1351	*1351	*222	5707[2]	
33601 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
33719 (10P)	*400	*258	*1056	*1056	*222	5687[2]	
33729 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
34103 (6P)	*400	*258	*290	*290	*222	5687[2]	
34103 (8P)	*400	*258	*751	*751	*222	5687[2]	
34103 (10P)	*400	*258	*1051	*1051	*222	5687[2]	
34103 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
34123 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
34123 (14P)	*222	*222	*1351	*1351	*222	5707[2]	
34123 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
34231 (10P)	*400	*258	*1056	*1056	*222	5687[2]	
34251 (10P)	*400	*258	*1053	*1053	*222	5687[2]	
37431 (12)	*400	*258	*1351	*1351	*222	5687[2]	
37431 (14P)	*222	*222	*1351	*1351	*222	5707[2]	
37431 (18P)	*85	*85	*1653	*1653	*85	5707[2]	
41783 (12P)	*400	*258	*1351	*1351	*222	5687[2]	
41803 (14P)	*222	*222	*1351	*1351	*222	5707[2]	
41803 (18P)	*85	*85	*1653	*1653	*85	5707[2]	

1 The 4019 printer has five resident fonts: FGID 11, 46, 85, 159, and 254. The AS/400 system sends any of those FGIDs that do not show a substitution in Table 83 to the emulator that the 4019 is attached to. The emulator may not support all of the FGIDs and may report an error or perform a substitution of its own.

2 The 4028 performs the font substitution as shown unless a font card has been installed that contains that FGID. For example, if a font card with an FGID of 2 is installed, the AS/400 system sends the FGID of 2 to the printer. However, if the font card is not installed, the AS/400 system substitutes an FGID of 11.

Font Substitution by Font ID Range

If your application specifies a font ID that is not found in Table 83 or is not resident in the printer (font card), the system makes a substitution based on the font ID ranges in the following table. For example, if font ID 4 is specified in your application, the AS/400 system substitutes font ID 11 as shown in the table below.

Table 84. Font Substitution by Font ID Range.

FGID			Substituted FGID
Fonts 0 through 65			11
Fonts 66 through 153			85
Fonts 154 through 200			160
Fonts 201 through 210			204
Fonts 211 through 239			223
Fonts 240 through 246			245
Fonts 247 through 257			252
Fonts 258 through 259			259
Fonts 260 through 273			434
Fonts 274 through 279			279
Fonts 280 through 284			281
Fonts 285 through 289			285
Fonts 290 through 299			290
Fonts 300 through 511			252
Fonts 512 through 2303			252
Fonts 2304 through 3839 or Fonts 4069 through 65279	Fonts with point size equal to 0		252
	Fonts with point size greater than 0 but less than 7.6		5687-6p
	Fonts with point size greater than or equal to 7.6 but less than 9.6		5687-8p
	Fonts with point size greater than or equal to 9.6 but less than 11.6		5687-10p
	Fonts with point size greater than or equal to 11.6 but less than 13.6		5687-12p
	Fonts with point size greater than or equal 13.6 but less than 17.6		5707-14p
	Fonts with point size greater than or equal to 17.6 but less than 23.6		5707-18p
	Fonts with point size greater than or equal to 23.6		5707-24p
Fonts 3840 through 4095 (User-defined)			No Substitution
Fonts 65280 through 65534 (User-defined)			No Substitution

Character Identifier (CHRID) Values Supported

The following table lists all the character identifiers, the related national language groups, the correct code page, and which printers support which character identifier.

Table 85 (Page 1 of 3). CHRID Values and Applicable Printers (CHRID Parameter)

	Code Pages		Printers[1]						
Language Groups	CHRID Code Page xxx yyy[2,3]	Sub-stitute Code Page yyy[2,4]	3812[5] 3816[5]	4214[5]	4224[5] 4230[5]	4234[5]	5219	5224 5225	4028[5]
Major Groups									
International (and US ASCII)	103 038	500	Yes	N/A	N/A	N/A	Yes	N/A	Yes
Multinational	697 500		Yes	Yes	Yes	Yes	N/A	N/A	Yes
	337 256	500	Yes	N/A	N/A	N/A	N/A	Yes	Yes
	697 256	500	Yes	N/A	Yes	IPDS[7]	N/A	N/A	Yes
United States	101 037		Yes	Yes	Yes	Yes	Yes	Yes	Yes
	697 037		Yes	Yes	N/A	Yes	N/A	N/A	Yes
Individual Countries/Languages									
Arabic	697 361		Yes	N/A	Yes	N/A	N/A	N/A	Yes
Arabic X/B	235 420	500	Yes	N/A	N/A	N/A	N/A	N/A	Yes
	697 420		Yes	N/A	N/A	IPDS[7]	N/A	N/A	Yes
Austria/Germany[6]	265 273		Yes	Yes	Yes	Yes	Yes	Yes	Yes
	697 273		Yes	Yes	Yes	Yes	N/A	N/A	Yes
Austria/Germany	697 286	273	Yes	N/A	Yes	N/A	N/A	N/A	Yes
Belgium[6]	697 500		N/A	Yes	Yes	Yes	Yes	Yes	N/A
	269 274		N/A	Yes	Yes	SCS[8]	Yes	Yes	N/A
	697 274		N/A	Yes	Yes	SCS[8]	N/A	N/A	N/A
Brazil[6]	273 275		Yes	Yes	Yes	Yes	Yes	Yes	Yes
	697 275		Yes	Yes	Yes	Yes	N/A	N/A	Yes
Canadian French[6]	277 276	297	Yes	N/A	N/A	N/A	Yes	Yes	Yes
	341 260	037	Yes	N/A	Yes	N/A	N/A	N/A	Yes
	697 260		Yes	N/A	N/A	N/A	N/A	N/A	Yes
Canada-Bilingual	038 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
	039 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Canada-English	037 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Cyrillic	960 880		N/A	N/A	Yes	IPDS[7]	N/A	N/A	Yes
Czechoslovakia/Czech	083 257		N/A	N/A	N/A	N/A	Yes	N/A	N/A
Czechoslovakia/Slovak	085 257		N/A	N/A	N/A	N/A	Yes	N/A	N/A
Denmark/Norway[6]	281 277		Yes	Yes	Yes	Yes	Yes	Yes	Yes
	697 277		Yes	Yes	Yes	Yes	N/A	N/A	Yes
Denmark/Norway	697 287	277	Yes	N/A	Yes	N/A	N/A	N/A	Yes
Finland/Sweden[6]	285 278		Yes	Yes	Yes	Yes	Yes	Yes	Yes
	697 278		Yes	Yes	Yes	Yes	N/A	N/A	Yes
Finland/Sweden	697 288	278	Yes	N/A	Yes	N/A	N/A	N/A	Yes
France (1977)[6]	289 279	297	Yes	N/A	N/A	N/A	N/A	Yes	Yes
France (1980)[6]	288 297		Yes	Yes	Yes	Yes	Yes	N/A	Yes
	697 297		Yes	Yes	Yes	Yes	N/A	N/A	Yes
France	251 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
France/Belgium	031 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Germany/Austria	028 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
	029 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Greek	218 423		N/A	N/A	Yes	IPDS[7]	N/A	N/A	Yes
	925 875		N/A	N/A	Yes	IPDS[7]	N/A	N/A	Yes
Hebrew	941 424		Yes	N/A	Yes	IPDS[7]	N/A	N/A	Yes
	697 424		Yes	N/A	Yes	IPDS[7]	N/A	N/A	Yes
Hong Kong	119 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Hungary	091 257		N/A	N/A	N/A	N/A	Yes	N/A	N/A
Icelandic	697 871		Yes	N/A	Yes	IPDS[7]	N/A	N/A	Yes

Table 85 (Page 2 of 3). CHRID Values and Applicable Printers (CHRID Parameter)

Language Groups	Code Pages: CHRID Code Page xxx yyy[2,3]	Code Pages: Substitute Code Page yyy[2,4]	Printers[1]: 3812[5] 3816[5]	4214[5]	4224[5] 4230[5]	4234[5]	5219	5224 5225	4028[5]
	697 029		Yes	N/A	N/A	N/A	N/A	N/A	Yes
Italy[6]	293 280		Yes	Yes	Yes	IPDS[7]	Yes	Yes	Yes
	697 280		Yes	Yes	Yes	Yes	N/A	N/A	Yes
Italy	041 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Japan-English[6]	297 281		Yes	Yes	Yes	IPDS[7]	Yes	Yes	Yes
	697 281		Yes	Yes	Yes	Yes	N/A	N/A	Yes
	068 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
	069 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Japan-Katakana[6]	332 290		Yes	N/A	Yes	Yes	N/A	Yes	Yes
Korean	933 833		N/A	N/A	4230-Yes 4224-N/A	N/A	N/A	N/A	N/A
	697 290		Yes	N/A	N/A	N/A	N/A	N/A	Yes
Latin	959 870		N/A	N/A	Yes	IPDS[7]	N/A	N/A	Yes
Latin America/Puerto Rico	025 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Netherlands	043 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Norway/Denmark	055 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Poland	093 257		N/A	N/A	N/A	N/A	Yes	N/A	N/A
Portugal[6]	301 282		Yes	Yes	Yes	Yes	Yes	Yes	Yes
	697 282		Yes	Yes	Yes	Yes	N/A	N/A	Yes
Portugal	697 831	282	Yes	N/A	Yes	N/A	N/A	N/A	Yes
	063 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Romania	087 258		N/A	N/A	N/A	N/A	Yes	N/A	N/A
South Africa	081 258		N/A	N/A	N/A	N/A	Yes	N/A	N/A
Spain[6]	305 283	284	Yes	N/A	Yes	Yes	Yes	Yes	Yes
	697 283	284	Yes	N/A	Yes	N/A	N/A	N/A	Yes
	697 289	284	Yes	N/A	Yes	N/A	N/A	N/A	Yes
	045 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Spanish Speaking[6]	309 284		Yes	Yes	Yes	Yes	Yes	Yes	Yes
	697 284		Yes	Yes	Yes	Yes	Yes	N/A	Yes
	149 284		N/A	N/A	N/A	N/A	Yes	N/A	
Sweden/Finland	052 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
	053 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Switzerland/French	048 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Switzerland/German	049 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Thai	1102 889		N/A	N/A	Yes	IPDS[7]	N/A	N/A	N/A
	938 838		N/A	N/A	4230-Yes 4224-N/A	N/A	N/A	N/A	N/A
Turkish	965 905		N/A	N/A	4230-No 4224-Yes	IPDS[7]	N/A	N/A	Yes
	1152 1026		N/A	N/A	4230-Yes 4224-N/A	N/A	N/A	N/A	N/A
United Kingdom[6]	313 285		Yes	Yes	Yes	Yes	Yes	Yes	Yes
	697 285		Yes	Yes	Yes	Yes	N/A	N/A	Yes
U.K./Israel	066 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
U.K./Israel-Latin	067 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
USA-Accounting	017 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
USA/Australia	001 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Countries of the former Yugoslavia	410 890		N/A	N/A	Yes	IPDS[7]	N/A	N/A	N/A
Countries of the former Yugoslavia-Latin	095 257		N/A	N/A	N/A	N/A	Yes	N/A	N/A
Noncountry Languages									
APL	697 293	Yes	Yes	N/A	N/A	N/A	N/A	N/A	Yes
APL Alternate	697 310		Yes	N/A	N/A	N/A	N/A	N/A	Yes

Table 85 (Page 3 of 3). CHRID Values and Applicable Printers (CHRID Parameter)

Language Groups	Code Pages: CHRID Code Page xxx yyy[2,3]	Code Pages: Substitute Code Page yyy[2,4]	Printers[1]: 3812[5] 3816[5]	4214[5]	4224[5] 4230[5]	4234[5]	5219	5224 5225	4028[5]
ASCII	103 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
DCF Compatibility	1132 1002		Yes	N/A	4230-Yes 4224-No	N/A	N/A	N/A	Yes
EBCDIC	101 256		Yes	N/A	N/A	N/A	Yes	N/A	Yes
International Typographic	697 361		Yes	N/A	N/A	N/A	N/A	N/A	Yes
OCR (unregistered)	697 340	500	Yes	N/A	Yes	IPDS[7]	N/A	N/A	Yes
OCR A	697 892	500	Yes	N/A	N/A	IPDS[7]	N/A	N/A	Yes
OCR A (unregistered)	580 340	500	Yes	N/A	Yes	IPDS[7]	N/A	N/A	Yes
OCR B	697 893	500	Yes	N/A	N/A	IPDS[7]	N/A	N/A	Yes
OCR B (unregistered)	590 340	500	Yes	N/A	Yes	IPDS[7]	N/A	N/A	Yes
Personal Computer	697 437		Yes	N/A	N/A	N/A	N/A	N/A	Yes
Symbol-Selectric	201 259	500	Yes	N/A	N/A	N/A	Yes	N/A	Yes
Symbol-6640	202 259	500	Yes	Yes	N/A	N/A	Yes	N/A	Yes
Symbol-6670	203 259		Yes	N/A	N/A	N/A	Yes	N/A	Yes
Symbols Set 7	697 259		Yes	N/A	N/A	N/A	N/A	N/A	Yes

1 The 5256, 5262, and 4245 work station printers do not support the hardware function required for alternative CHRID processing. If a nondefault character set and code page is selected for these printers, a diagnostic message is sent and processing continues using the default character set.

2 If the printer supports the code page specified (the second part *(yyy)* of the CHRID parameter) but not the character set *(xxx)*, then the character set supported by the printer is used along with the specified code page. For example, if 337 037 (extended character set for displays) is specified for the 5224 and 5225 Printers, the print file is printed with character set 101, code page 037.

3 In some cases, the printer will substitute a supported code page for an unsupported code page. Consult the various printer reference guides for defaults on the code page mapping.

4 If the printer does not support or map the code page specified, an attempt is made by the system to find a satisfactory substitute. This column shows the code page substitutes that are made if the specified printer supports the substitute.

5 The 3812, 3816, 4214, 4224, 4230, and 4234 Printers support character set 697 (full character set). This character set contains all the characters in the limited character sets. For example, 697 037 would contain all the characters in 101 037 or 337 037 (extended character set for displays).

6 This language is considered a primary language group. All other entries, if any, under the primary language group are considered as alternative language groups.

7 Supported by 4234 IPDS version only.

8 Supported by 4234 SCS version only.

Lines Per Inch (LPI) Values Supported

Lines per inch means the number of characters that can be printed vertically within an inch.

Each entry in the following table shows the valid range of values for lines per page for each printer type and for each value of lines per inch (LPI) valid for the printer.

Table 86. Lines per Page (LPI Parameter)

Printer	3 Lines per Inch	4 Lines per Inch	6 Lines per Inch	7.5 Lines per Inch	8 Lines per Inch	9 Lines per Inch	12 Lines per Inch
3287	–	1-104	1-104	–	1-104	–	–
3812 SCS	–	1-56	1-84	–	1-112	1-126	1-168
3812 IPDS	–	2-56	2-84	–	2-112	2-112	2-168
3816 SCS	–	1-56	1-84	–	1-112	1-126	1-168
3816 IPDS	–	2-56	2-84	–	2-112	2-112	2-168
3820	–	1-56	1-84	–	1-112	1-126	1-168
3825	–	1-56	1-84	–	1-112	1-126	1-168
3827	–	1-56	1-84	–	1-112	1-126	1-168
3835	–	2-91	2-136	–	2-182	2-204	2-273
4028	–	2-56	2-84	–	1-112	1-112 or 2-126	2-168
4214	–	1-255	1-255	–	1-255	1-255	–
4224, 4234 IPDS	–	2-91	2-136	–	2-182	2-204	2-273
4230	–	2-91	2-136	–	2-182	2-204	2-273
4234 SCS	–	1-255	1-255	–	1-255	–	–
4245 Models T12 and T20	–	–	1-255	–	1-255	–	–
5211	–	–	2-84	–	2-112	–	–
5219 Continuous Forms	–	2-255	2-255	–	2-255	–	2-255
5219 Cut Sheet	–	57	86	–	114	–	172
5224	–	1-255	1-255	–	1-255	1-255	–
5225	–	1-255	1-255	–	1-255	1-255	–
5256 (set manually)	–	–	1-255	–	1-255	–	–
5262	–	–	1-255	–	1-255	–	–
5553	1-255	1-255	1-255	1-255	1-255	–	1-255
5583	1-255	1-255	1-255	1-255	1-255	–	–
6252	–	1-255	1-255	–	1-255	1-255	–

Characters Per Inch (CPI) Values Supported

Characters per inch means the number of characters printed horizontally within an inch across a page.

Each entry in the following table shows the valid range of values for the characters per line for each printer type and for each value of characters per inch (CPI) for the printer.

Table 87. Characters per Line (CPI Parameter)

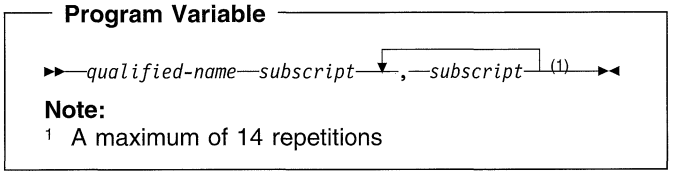
Printer	5 Characters per Inch	10 Characters per Inch	12 Characters per Inch	13.3 Characters per Inch	15 Characters per Inch	16.7 Characters per Inch	18 Characters per Inch	20 Characters per Inch
3287	–	1-132	–	–	–	–	–	–
3812[1]	1-42	1-85	1-102	–	1-127	–	–	–
3812[1] Rotated Form	1-70	1-140	1-168	–	1-210	–	–	–
3816[1]	1-42	1-85	1-102	–	1-127	–	–	–
3816[1] Rotated Form	1-70	1-140	1-168	–	1-210	–	–	–
3820[1]	–	1-85	1-102	–	1-127	–	–	–
3825[1]	–	1-85	1-102	–	1-127	–	–	–
3827[1]	–	1-85	1-102	–	1-127	–	–	–
3835[1]	–	1-132	1-158	–	1-198	–	–	–
4214 Continuous Forms	1-66	1-132	1-158	–	1-198	1-220	–	–
4214 Cut Sheet	1-60	1-120	1-144	–	1-180	1-200	–	–
4224[1]	–	1-132	1-158	–	1-198	1-220	–	–
4028[1]	1-42	1-85	1-102	–	1-127	–	–	–
4028[1] Rotated Form	1-70	1-140	1-168	–	1-210	–	–	–
4230[1]	–	1-132	1-158	–	1-198	1-220	–	–
4234 SCS[1]	–	1-132	–	–	1-198	–	–	–
4234 IPDS[1]	–	1-132	–	–	1-198	1-238	–	–
4245	–	1-132	–	–	–	–	–	–
5219	–	1-132	1-158	–	1-198	–	–	–
5224	–	1-132	–	–	1-198	–	–	–
5225	–	1-132	–	–	1-198	–	–	
5256 Model 3	–	1-132	–	–	–	–	–	–
5262	–	1-132	–	–	–	–	–	–
5553	–	1-136	1-163	1-181	1-204	–	1-244	1-272
5583	–	1-132	1-158	1-176	1-198	–	1-236	1-264
6252	–	1-132	–	–	1-198	–	–	–

1 Many character per inch values (implied by the pitch of the font, see the FONT parameter), are supported in addition to the ones listed here. To find the maximum characters per line, multiply the implied characters per inch value listed in the font table by maximum page width supported (in inches). The maximum page width supported by the 3812 and 3816 Printers is 8.5 inches for non-rotated forms and 14.0 inches for rotated forms.

Appendix C. Parameter Values Used for Testing and Debugging

This appendix contains expanded descriptions of the program variable, basing pointer, subscript, and qualified-name parameter values. These values can be specified on the Add Breakpoint (ADDBKP), Add Trace (ADDTRC), Change High-Level Language Pointer (CHGHLLPTR), Change Program Variable (CHGPGMVAR), Change Pointer (CHGPTR), and Display Program Variable (DSPPGMVAR) commands.

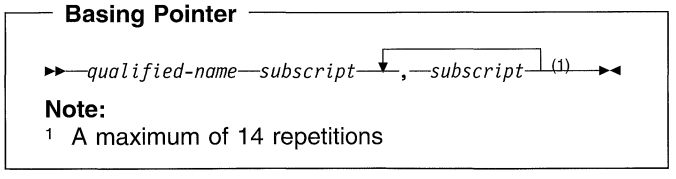
Program-Variable Description



The program variable must be enclosed in apostrophes if it contains special characters. Up to 132 characters can be specified for a program variable name. This includes any subscripts, embedded blanks, parentheses, and commas. It does not include the enclosing apostrophes when special characters are used. Some examples are:

```
COUNTA
'VAR1(2,3)'
'A.VAR1(I,3,A.J,1)'
'VAR1 OF A(I,3,J OF A)'
'&LIBNAME'
```

Basing-Pointer Description



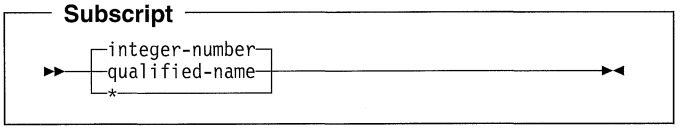
The basing pointer must be enclosed in apostrophes if it contains special characters. Up to 132 characters can be specified for a basing pointer name. This includes any subscripts, embedded blanks, parentheses, and commas. It does not include the enclosing apostrophes when special characters are used. Some examples are:

```
PTRVAR1
'ABC.PGMPTR(5,B.I)'
```

If more than one basing pointer is specified for a variable, the list of basing pointers must be enclosed in parentheses. When multiple basing pointers are specified, they must be listed in order, from the first basing pointer to the last, when used to locate the variable. In the example below, the PTR_1 basing pointer is the first basing pointer used to locate the variable; it either must have a declared basing pointer, or it must not be a based variable. The address contained in the PTR_1 pointer is used to locate the A.PTR_2 pointer (which must be declared as a based pointer variable). The contents of the A.PTR_2 pointer are used to locate the PTR_3 pointer array (which must also be declared based), and the contents of the specified element in the last pointer array are used to locate the variable. An example is:

```
('PTR_1' 'A.PTR_2' 'PTR_3(1,B.J)')
```

Subscript Description



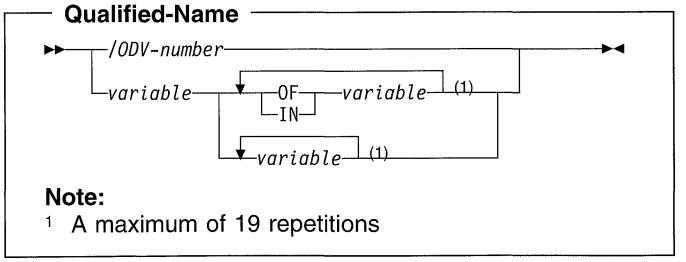
An integer number contains from 1 through 15 digits with an optional leading sign (either plus or minus). A decimal point is not allowed in an integer-number subscript. If a decimal point is specified, the subscript value is not interpreted as the correct numeric value by the system, and an error message is returned.

An asterisk (*) can be used to request a single-dimensional cross-section display of an array program variable. An asterisk can only be specified for a subscript on the primary variable (not on a basing pointer) for the PGMVAR keyword on the Add Break Point (ADDBKP), Add Trace (ADDTRC), and Display Program Variable (DSPPGMVAR) commands. In addition, if the variable has multiple dimensions, only one of the subscript values can be an asterisk. An example of a request to display an array cross-section is:

```
DSPPGMVAR  PGMVAR('X1(*,5,4)')
```

This display shows the values of all elements of the array that have the second subscript equal to five, and the third subscript equal to four.

Qualified-Name Description



Note: Some high-level languages may allow you to declare more than one variable with the same fully qualified name (although you generally are not able to refer to these variables in the high-level language program after they are declared). If you attempt to refer to

such a variable using an OS/400 test facility command, the system selects one of the variables and uses it for the operation. No error is reported when a duplicate fully qualified name is selected.

Rules for Qualified Name Description

- An ODV number is a slash (/) followed by 1 to 4 hexadecimal digits (0 through 9, and A through F).
- The variable-name must be the name of a variable in the program. This name must be specified the same way in the high-level language. Some high-level languages introduce qualifier variable names in addition to the ones you specified in the source for your program. See the appropriate high-level language manual for more information about variable names.
- Blanks must separate the variable-names from the special words OF and IN.
- When a period is used to form a qualified name, no blanks can appear between it and the variable-names.
- The ordering of the variable names must follow these rules:
  - For qualified names that contain no embedded period, the variable names are assumed to be specified from the lowest to the highest levels in the structure.
  - For qualified names that contain one or more embedded periods, the variable names are assumed to be specified from the highest to the lowest levels in the structure.
- When an ODV number is not used for the qualified name, enough qualifier variable names must be specified so that a single unique variable can be identified in the program. Whether the qualified name is a simple name (only one variable name specified) or a name with multiple qualifier variable names, the variable in the program is uniquely identified if either of the following conditions is true (these conditions may require you to specify more qualifier variable names for OS/400 test facility commands than you need to specify in the high-level language program to uniquely select a program variable):
  - A variable is uniquely identified if there is one and only one variable in the program with a set of qualifier variables matching the qualified variable name specified.
  - A variable is uniquely identified in the program if it has exactly the same set of qualifier variables as the qualifier variable names specified. When the complete set of qualifiers is specified, the variable name is said to be *fully qualified*. A variable that is a *fully qualified* match for the qualified-name is selected even if there are other variables with names that match the qualified name but have additional qualifier variables which were not specified.

Glossary

This glossary includes terms and definitions from:

- The *American National Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 1430 Broadway, New York, New York 10018. Definitions are identified by the symbol (A) after the definition.
- The *Information Technology Vocabulary*, developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Committee (ISO/IEC JTC1/SC1). Definitions of published parts of this vocabulary are identified by the symbol (I) after the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.

access method. A method used to read a record from, or to write a record into, a file. Access can be sequential (records are processed one after another in the order in which they appear in the file), it can be random (the individual records can be processed in any order), or it can be dynamic (records can be processed sequentially or randomly, depending on the specific request).

access path. The order in which records in a database file are organized for processing by a program.

access path journaling. A method of recording changes to an access path as changes are made to the data in the database file so that the access path can be recovered automatically by the system.

accounting code. A 15-character field, assigned to a job by the system when it is processed by the system, that is used to collect statistics for the system resources used for that job when job accounting is active.

accounting segment. The period of time during which statistics are gathered, beginning when the job starts or when the job's accounting code is changed, and ending when the job ends or when the job's accounting code is next changed.

acknowledgment. In BSC, a positive response to a data transmission.

acquire. To assign a display station or session to a program.

activation group. A substructure of a run-time job. An activation group consists of system resources (storage for program or procedure variables, commitment definitions, and open files) allocated to one or more programs. An activation group is like a miniature job within a job.

active file. A file on a tape or diskette volume with an expiration date later than the system date.

activity level. A characteristic of a subsystem that specifies the maximum number of jobs that can compete at the same time for the processing unit.

ADCS. See *IBM Advanced Data Communications for Stores (ADCS)*.

add authority. A data authority that allows the user to add entries to an object; for example, add job entries to a job queue or add records to a file. Contrast with *delete authority*. See also *read authority* and *update authority*.

adjacent node. In OSI, a node that is attached to the same subnetwork as the local node. An adjacent node can be either a destination node or a relay node.

adopted authority. Authority given to the user by the program for the duration of the job that uses that program. The program must be created with owner authority.

Advanced Function Printing (AFP). The ability of programs to use the all-points-addressable concept to print text and images on a printer.

Advanced Peer-to-Peer Networking (APPN). Data communications support that routes data in a network between two or more APPC systems that do not need to be adjacent.

advanced printer function (APF). A function of the AS/400 Application Development Tools licensed program that allows a user to design symbols, logos, special characters, large characters, and forms tailored to a business or data processing application. The function supports printing of any design on the 5224 or 5225 dot matrix printer.

advanced program-to-program communications (APPC). Data communications support that allows programs on an AS/400 system to communicate with programs on other systems having compatible communications support. APPC on the AS/400 system provides an application programming interface to the SNA LU type 6.2 and node type 2.1 architectures.

AFP. See *Advanced Function Printing (AFP)*.

AFP resources. The form definitions, page definitions, fonts, overlays (electronic forms), and page segments (graphic images). With PrintManager*, resources can either exist in a system library, or be placed inline with a print job as the job is written to the spool.

AID. See *attention identifier (AID)*.

alert. In SNA, a record sent to a focal point to identify a problem or an impending problem.

alert controller description. A controller description that defines the system to which alerts will be sent on an alert controller session.

alert controller session. A type of SSCP-PU session on which alerts can be sent to a system that is designated as an alert focal point.

alert description. Information in an alert table that defines the contents of a Systems Network Architecture (SNA) alert for a particular message ID.

alert focal point. The system in a network that receives and processes (logs, displays, and optionally forwards) alerts. An alert focal point is a subset of a problem management focal point.

alert table. An object consisting of alert descriptions that define the contents of a Systems Network Architecture (SNA) alert for particular error conditions. The system-recognized identifier for the object type is *ALRTBL.

all authority. An object authority that allows the user to perform all operations on the object except those limited to the owner or controlled by authorization list management authority. The user can control the object's existence, specify the security for the object, and change the object. Contrast with *exclude authority*.

all object authority. A special authority that allows users to use all system resources without having specific authority to the resources.

alphanumeric. Pertaining to the letters, A through Z or a through z; numbers, 0-9; and special symbols, $, #, @, ., or _.

alphanumeric character. In COBOL, any character in the character set of the computer.

alternative collating sequence. A user-defined collating sequence that replaces the standard EBCDIC collating sequence.

American National Standards Institute (ANSI). An organization sponsored by the Computer and Business Equipment Manufacturers Association for establishing voluntary industry standards.

ANSI. See *American National Standards Institute (ANSI)*.

APAR. See *authorized program analysis report (APAR)*.

APPC. See *advanced program-to-program communications (APPC)*.

application association. In OSI, a cooperative relationship between two application entities that enables them to exchange data.

| **application developer**. In the SAA Application Development Manager/400 product, an application programmer who uses the SAA Application Development Manager/400 environment to develop code.

application layer. In systems interconnection architecture, the layer of the model that provides a means for application processes residing in different systems to exchange information.

| **APPN**. See *Advanced Peer-to-Peer Networking (APPN)*.

arrival sequence. An order in which records are retrieved that is based on the order in which records are stored in a physical file.

arrival sequence access path. An access path to a database file that is arranged according to the order in which records are stored in the physical file.

ascending key. The values by which data is arranged from the lowest value to the highest value of the key field in accordance with the rules for comparing data items.

ascending sequence. The arrangement of data in order from the lowest value to the highest value, according to the rules for comparing data.

assistance level. The type of displays that a user selects to interact with the system. The three levels of assistance available are basic, intermediate, and advanced.

| **ASP**. See *auxiliary storage pool (ASP)*.

asynchronous. Not occurring in a regular or predictable pattern.

asynchronous balanced mode (ABM). In communications, an operational mode of a balanced data link in which either combined station can send commands at any time and can initiate transmission of response frames without explicit permission from the other combined station.

asynchronous communications. A method of communications supported by the operating system that allows an exchange of data with a remote device, using either a start-stop line or an X.25 line. Asynchronous communications includes the file transfer support and the interactive terminal facility support.

| **attention identifier (AID)**. A character in a data stream that is sent to the host system when a display station user presses an attention identifier (AID) key. Typical AID keys are function keys or the Clear, Enter, Page Up, Page Down, Help, Print, and Home keys.

| **attribute**. (1) A characteristic or trait of one or more items. (2) In user interface manager (UIM) tag language, an identifier used with related material that takes on a specific meaning, such as an action to be taken or the characteristics of text or data.

authority checking. A function of the system that looks for and verifies a user's authority to an object.

authority holder. An object that specifies and reserves an authority for a program-described database file before the file is created. When the file is created, the authority specified in the holder is linked to the file.

authorization ID. In SQL, a user profile. A name identifying a user to whom privileges can be granted.

authorization list. A list of two or more user IDs and their authorities for system resources. The system-recognized identifier for the object type is *AUTL.

authorization list management authority. An object authority that allows the user to add users to, remove users from, and change users' authorities on the authorization list.

authorized program analysis report (APAR). A request for correction of a defect in a current release of an IBM-supplied program.

automatic answer. In data communications, a line type that does not require operator action to receive a call over a switched line.

automatic bind. In SQL, the bind that automatically takes place when an application program is run and the bound access plan is nullified; that is, without a user issuing a CRTSQLxxx command (where *xxx* is C, CBL, FTN, PLI, or RPG). See also *bind* and *dynamic bind*.

automatic call. A feature that permits a station to connect with another station over a switched line without operator action.

automatic call unit. A common carrier device that allows the AS/400 system to automatically dial a remote location.

automatic configuration. A function that names and creates the descriptions of network devices and controllers attached to a preexisting line. The objects are also varied on at a user's request.

automatic data. Data that is allocated with the same value on entry and reentry into a procedure. Values of the data on exiting from the procedure are not retained for the next entry into the procedure. The scope of automatic data is a procedure call within an activation group. Contrast with *external data* and *local data*.

automatic vary on. An option specified during the creation of configuration objects that allows them to be available when the system is started (IPL).

autostart. (1) Pertaining to a system activity that starts automatically, usually based on the start or end of some other activity. (2) An OSI Communications Subsystem/400 function that starts an X.25 line automatically when the line set that it belongs to is started.

autostart job. A job doing repetitive work or one-time initialization work that is associated with a particular subsystem. The autostart jobs associated with a subsystem are automatically started each time the subsystem is started.

auxiliary storage. All addressable disk storage other than main storage.

auxiliary storage pool (ASP). One or more storage units defined from the disk units or disk unit subsystems that make up auxiliary storage. ASPs provide a means of isolating certain objects on specific disk units to prevent the loss of data due to disk media failures on other disk units. See also *unit*, *system ASP*, and *user ASP*.

B-channel. In ISDN, a duplex channel for transmitting data or digital voice across the network.

backup. (1) Pertaining to an alternative copy used as a substitute if the original is lost or destroyed, such as a backup log. (2) The act of saving some or all of the objects on a system to a tape, diskette, or save file. (3) The tapes, diskettes, or save files with the saved objects. (4) For communications, see *switched network backup (SNBU)*.

base pool. A storage area that contains all unassigned main storage on the system and whose minimum size is specified in the system value QBASPOOL. The system-recognized identifier is *BASE.

basic conversation. In APPC, a temporary connection between an application program and an APPC session in which the user must provide all the information on how the data is formatted.

basic data exchange. A file format for exchanging data on diskettes or tape between systems or devices.

basic rate interface (BRI). In ISDN, an interface that provides two 64 000 bps data channels (B-channels) and one 16 000 bps signaling channel (D-channel). Also known as *2B + D*.

batch job. A predefined group of processing actions submitted to the system to be performed with little or no interaction between the user and the system.

batch subsystem. A part of main storage where batch jobs are processed.

beaconing. Pertaining to an adapter in a token-ring network that repeatedly sends a frame (beacon message) when it is not receiving a normal signal because of serious error, such as a line break or power failure. The message frame repeats until the error is corrected or bypassed.

binary synchronous communications (BSC). A data communications line protocol that uses a standard set of transmission control characters and control character sequences to send binary-coded data over a communications line.

binary synchronous communications equivalence link (BSCEL) support. The intersystem communications function (ICF) support on the AS/400 system that provides binary

synchronous communications with another AS/400 system, System/36, System/38, and many other BSC computers and devices.

bind. (1) In SQL, the process by which the output from the SQL precompiler is converted to a usable structure called an access plan. This process is the one during which access paths to the data are selected and some authorization checking is performed. See also *automatic bind* and *dynamic bind*. (2) To create a program, which can be run, by combining one or more modules created by an Integrated Language Environment (ILE) compiler. See also *binder* and *binding*.

binder. The system component that creates a bound program by packaging Integrated Language Environment (ILE) modules and resolving symbols passed between those modules.

binding directory. A list of modules and service programs. A binding directory is not a repository of those objects; instead, it allows them to be referred to by name and type.

block. A group of records that are recorded or processed as a unit.

bound program. An AS/400 object that combines one or more modules created by an Integrated Language Environment (ILE) compiler. See also *service program*.

bps. Bits per second.

bracket. In SNA, one or more chains of request units and their responses, representing a complete transaction, exchanged between two logical unit (LU) half-sessions.

break delivery. The method of delivering messages to a message queue in which the job associated with that message queue is interrupted as soon as the message arrives.

breakpoint. A stopping place in a statement of a high-level language.

bridge. A device that connects two or more networks; for example, an Ethernet-to-Ethernet network or Ethernet to token-ring network. A bridge stores and forwards information in packets between the networks.

BSC. See *binary synchronous communications (BSC)*.

BSCEL support. See *binary synchronous communications equivalence link (BSCEL) support*.

built-in function. (1) In PL/I and CL, a predefined function, such as a commonly used arithmetic function or a function necessary to high-level language compilers (for example, a function for manipulating character strings or converting data). It is automatically called by a built-in function reference. (2) In REXX, a function that is supplied by a language. These functions, defined as part of the REXX language, include character manipulation, conversion, and information functions.

burst. In AFP support, to separate continuous-forms paper into separate sheets.

bus. One or more conductors used for transmitting signals or power.

business graphics. See *graphics*.

Business Graphics Utility (BGU). See *IBM AS/400 Business Graphics Utility Version 2 (BGU)*.

C & SM. See *communications and systems management (C & SM)*.

call. In telephony, a physical or logical connection (association) between one or more parties in a telephone call. For example, a held call has two parties logically connected although they are physically disconnected. A partial call is a two-party call in which one of the two parties is a virtual party; this can be viewed as a transient stage of the telephone call.

call control. That set of telephony functions that includes call establishment, call transfer, and call disconnection (the program control of a telephone call).

call level. The position of an entry (program or procedure) in the call stack. The first entry has a call level of 1. Any entry called by a level 1 entry has a call level of 2, and so on.

call stack. The ordered list of all programs or procedures currently started for a job. The programs and procedures can be started explicitly with the CALL instruction, or implicitly from some other event.

call stack entry. A program or procedure in the call stack.

called program. A program that is the object of a CALL statement combined at run time with the calling program to produce a run unit.

caller. The requester of a service.

capacity planner. A function that uses information about the system, such as a description of the system's workload, performance objectives, and configuration, to determine how the data processing needs of the system can best be met. The capacity planner then recommends, through the use of printed reports and graphs, ways to enhance performance, such as hardware upgrades, performance tuning, or system configuration changes.

carrier. A continuous frequency that can be varied with a second signal to send information.

CCITT. The International Telegraph and Telephone Consultative Committee.

chain. (1) A group of logically linked records. (2) In DFU, a way to change from one display format to another after the user signals that the first display format was completed. (3) In SNA, a group of logically linked records that are transferred over a communications line. See also *RU chain*.

chaining. A method of storing records in which each record belongs to a list or group of records and has a linking field for tracing the chain.

change authority. An object authority that allows a user to perform all operations on the object except those limited to the owner or controlled by object existence authority and object management authority. The user can add, change, and delete entries in an object, or read the contents of an entry in the object. Change authority combines object operational authority and all the data authorities.

character. Any letter, number, or other symbol in the data character set that is part of the organization, control, or representation of data.

character codes. In PC Support/400, the ASCII or EBCDIC values assigned to the symbols or functions that are used by a computer.

character constant. The actual character value (a symbol, quantity, or constant) in a source program that is itself data, instead of reference to a field that contains the data.

character field. An area that is reserved for information that can contain any of the characters in the character set.

character format. In REXX, a format that is used in the REXX conversion functions to indicate that data is in a textual form as opposed to machine-readable form.

character key. A keyboard key that allows the user to type into the system the character shown on the key.

character variable. Character data whose value is assigned or changed while the program is running.

characters per inch (cpi). The number of characters printed horizontally within an inch across a page.

chart. In AS/400 Business Graphics Utility, displayed, printed, or plotted output that compares one or more sets of variable data in chart form. The types of charts are bar, line, pie, surface, histogram, Venn diagram, and text.

chart format. In AS/400 Business Graphics Utility, an object containing chart characteristics, such as the chart type, chart heading, legend position, and so on. The chart format does not include the data values to be plotted. The system-recognized identifier for the object type is *CHTFMT.

check out. In the SAA Application Development Manager/400 product, to copy a part to a specific development group, if it is not there already, and to set the access key for the part so no other application developer can change it.

checksum protection. A function that protects data stored in the system auxiliary storage pool from being lost because of the failure of a single disk. When checksum protection is in effect and a disk failure occurs, the system automatically reconstructs the data when the system program is loaded after the device is repaired.

CL. See *control language (CL)*.

class attributes. The values in a class object that control the processing of routing steps in a job. These values include the run priority, time slice, eligibility for purge, default wait time, maximum processing unit time, and maximum temporary storage parameters.

class object. An object that identifies the run attributes of a job. The system-recognized identifier for the object type is *CLS.

class-of-service description. A system object created for Advanced Peer-to-Peer Networking (APPN) that provides the information required to assign relative priority to the transmission groups and intermediate routing nodes for an APPN session. The system-recognized identifier for the object type is *COSD.

CLEAR. In SNA, a command used to delete all requests and responses related to the active session.

client. Any program that communicates with TCP/IP Connectivity Utilities/400 or uses its services.

CLNS path. In OSI, a path used when the connectionless-mode network service is used. Each CLNS path names data terminal equipment (DTE) to be used for outbound communication.

close. The function that ends the connection between a file and a program, and ends the processing.

cluster. In SNA, a group of stations that consist of a controller (cluster controller) and the work stations attached to it.

code page. (1) A particular assignment of hexadecimal identifiers to graphic characters. (2) In AFP support, a font file that associates code points and graphic character identifiers.

code point. (1) One of the bit patterns assigned to a character in a character set. On the AS/400 system, a code point is represented by a hexadecimal number. For example, in code page 256 (EBCDIC), the letter "e" is assigned a code point of hex 85. (2) An identifier in an alert description that represents a short unit of text. The code point is replaced with the text by an alert display program. (3) In AFP support, an 8-bit binary number representing one of 256 potential characters. (4) For SNA alerts, a 1-or 2-byte hexadecimal code that designates a particular piece of text to be displayed at the focal point.

command definition. An object that contains the definition of a command (including the command name, parameter descriptions, and validity-checking information) and identifies the program that performs the function requested by the command. The system-recognized identifier for the object type is *CMD.

command line. The blank line on a display where commands, option numbers, or selections can be entered.

command processing program (CPP). A program that processes a command. This program performs some validity checking and processes the command so that the requested function is performed.

command string. In query management, a character string that contains a query command.

commit. (1) To make all changes permanent that were made to one or more database files since the last commit or rollback operation, and make the changed records available to other users. (2) In SQL, the process that allows data changed by one application or user to be used by other applications or users. When a commit operation occurs, the locks are released to allow other applications to use the changed data.

commit cycle. The sequence of changes made between commitment boundaries.

commitment control. (1) A means of grouping database file operations that allows the processing of a group of database changes as a single unit through the Commit command or the removal of a group of database changes as a single unit through the Rollback command. (2) In FORTRAN, a means of grouping file operations that allows the processing of a group of database changes as a single unit or the removal of a group of database changes as a single unit.

Common Programming Interface (CPI). In the Systems Application Architecture* (SAA*) solution, a set of software interfaces, conventions, and protocols that provide a framework for writing applications with cross-system consistency.

communications adapter. A part that electrically or physically connects a computer or device to a data communications network.

communications and systems management (C & SM). A part of the system that contains the remote management support (also referred to as DHCF), the change management support (referred to as DSNX), and the problem management support (referred to as alerts).

communications configuration. The physical placement of communications controllers, the attachment of communications lines, and so forth; and the configuration descriptions that describe the physical configuration to the system and describe how the configuration will be used by the system. See also *line configuration*, *controller configuration*, and *device configuration*.

communications controller. The I/O processor card in the card enclosure.

communications line. The physical link (such as a wire or a telephone circuit) that connects one or more work stations to a communications controller, or connects one controller to another. Contrast with *data link protocol*.

communications side information. In CPI Communications, an object that contains initialization parameters, such as the name of the partner program with which a program can establish a conversation and the name of the logical unit (LU) at the partner program's node, which CPI Communications requires to establish a conversation. The system-recognized identifier for the object type is *CSI.

communications type. A method for application programs to communicate on a local AS/400 system, or between a local AS/400 system and a remote system using the intersystem communications function (ICF). Examples of these communications methods include (a) asynchronous communications, (b) binary synchronous communications (BSC), (c) finance communications. (d) intrasystem communications, (e) retail communications, and (f) Systems Network Architecture (SNA), such as advanced program-to-program communications (APPC) and SNA upline facility (SNUF).

compact. To replace repetitive bits in a file or folder with control bits so that the file or folder takes up less space when saved.

compaction. A function that removes repetitive bits from the data being processed and replaces the repetitive bits with control bits. Compaction reduces the amount of storage space required for the data.

compatibility. Ability to work in the system or ability to work with other devices or programs.

compatible. Pertaining to the characteristics that make devices, programs, products, or systems work together.

completion message. A message that tells the operator when work is successfully ended.

compress. To replace repetitive characters in a file or folder with control characters so that the file or folder takes up less space.

compression. A function that removes repetitive characters, spaces, or strings of characters from the data being processed and replaces the repetitive characters with control characters. Compression reduces the amount of storage space required for the data.

configuration. The physical and logical arrangement of devices and programs that make up a data processing system. See also *communications configuration*, *line configuration*, *controller configuration*, and *device configuration*.

configuration list. A list of local or remote locations, network addresses, or pass-through device descriptions used

by some types of communications descriptions. The system-recognized identifier for the object type is *CFGL.

configure. (1) To describe the interconnected arrangement of the devices, programs, communications, and optional features installed on a system. (2) To describe setting up auxiliary storage pools and checksum protection.

confirmation of delivery. The automatic notification to the sender of a message, note, or document as to when action is taken on the message, note, or document. Confirmation of delivery must be requested by the sender.

connection list. An AS/400 communications object for ISDN that provides a list of information used to determine when to accept incoming calls and what information to send with outgoing calls. The system-recognized identifier for the object type is *CNNL.

connection network. A switched network (such as a local area network, X.25, or public-switched dial network) that allows a local node to establish APPN connections to more than one undefined adjacent node.

console. A display station from which an operator can control and observe the system operation. For example, an operator can install the operating system, do an attended IPL, or sign on the system after using the End System (ENDSYS) command.

contention state. In data communications, a type of half-duplex line or data link control in which either user may transmit any time the line or link is available. If both users attempt to transmit at the same time, the protocols or the hardware determines who goes first.

contextual search. In SAA OfficeVision/400, a type of search that allows the user to find smaller text strings that are part of larger search fields in filed documents or personal directories.

control block. A storage area used by a program to hold control information.

control language (CL). The set of all commands with which a user requests system functions.

control language (CL) program. A program that is created from source statements consisting entirely of control language commands.

control language (CL) variable. A program variable that is declared in a control language program and is available only to the CL program.

control panel. A panel located on the processing unit on the front of the rack that contains lights and switches to operate or service the system.

control point (CP). A collection of tasks, which provide directory and route selection functions for Advanced Peer-to-Peer Networking (APPN). An end node control point provides its own configuration, session, and management services with assistance from the control point in its serving network node. A network node control point provides session and routing service.

control statement. In programming languages, a statement that is used to interrupt the continuous sequential processing of programming statements; for example, a conditional statement such as IF, PAUSE, or STOP.

control station. The controlling or primary computer on a multipoint line. The control station controls the sending and receiving of data. See also *host system*.

control storage. Storage in the computer that contains the programs used to control input and output operations and other machine operations. See also *auxiliary storage*. Contrast with *main storage*.

controller. A device that coordinates and controls the operation of one or more input/output devices (such as work stations) and synchronizes the operation of such devices with the operation of the system as a whole.

controller configuration. The process of creating configuration descriptions for the local (device configuration) and remote (communications configuration) controllers that make up a data processing system.

controller description. An object that contains a description of the characteristics of a controller that is either directly attached to the system or attached to a communications line. The system-recognized identifier for the object type is *CTLD.

controlling subsystem. The interactive subsystem that is automatically started first when the system is started and through which the system operator controls the system.

conversation. In APPC, the communications between the application program and another application program at the remote system.

country identifier (country ID). The country associated with the language a document is written in.

creation date. The system date when an object is created. See also *job date*, and *system date*.

cross-reference listing. The part of the compiler listing that tells where files, fields, and indicators are defined, referred to, and changed in a program.

current. In the SAA Application Development Manager/400 product, pertaining to a part that is built with the latest version of all the source and related parts used to create it. Contrast with *stale*.

current form. In query management, the form being applied against the data to produce the report being displayed or printed.

current library. The library that is specified to be the first user library searched for objects requested by a user. The name for the current library can be specified on the Sign-On display or in a user profile. When you specify an object name (such as the name of a file or program) on a command, but do not specify a library name, the system searches the libraries in the system part of the library list, then searches the current library before searching the user part of the library list. The current library is also the library that the system uses when you create a new object, if you do not specify a library name.

current position. In GDDM, the position, in user coordinates, that becomes the starting point for the next graphics routine, if that routine does not explicitly specify a starting point.

current release. The latest available release of the system that replaced the licensed internal code and/or the operating system.

D-channel. In ISDN, a common channel used for signaling and management of the network.

data area. A system object used to communicate data, such as CL variable values between the programs within a job and between jobs. The system-recognized identifier for the data area is *DTAARA.

data authority. A specific authority to read, add, update, or delete data.

data circuit-terminating equipment (DCE). The equipment installed at the user's premises that provides all the functions required to establish, maintain, and end a connection, and the signal conversion and coding between the data terminal equipment and the line.

data communications. The sending and receiving of data between computers and/or remote devices according to selected protocols.

data compression. The reduction of data volume on the media when performing save operations.

data country code (DCC). A 3-digit code, unique to each country, that specifies the X.21 call format used by a network in its International Data Number to call another station.

data decompression. Reconstruction of data from a compressed format when performing a restore operation.

data definition. In IDDU, information that describes the contents and characteristics of a field, record, or file.

data description specifications (DDS). A description of the user's database or device files that is entered into the system in a fixed form. The description is then used to create files.

data dictionary. In IDDU, an object for storing field, record format, and file definitions. The system-recognized identifier for the object type is *DTADCT.

data file. A group of related data records organized in a specific order. A data file can be created by the specification of FILETYPE(*DATA) on the create commands. Contrast with *source file*.

data file utility (DFU). The part of the AS/400 Application Development Tools licensed program that is used to enter, maintain, and display records in a database file.

data flow control layer. In SNA, the layer within a half-session that (a) controls whether the half-session can send or receive, or both send and receive request units (RUs) at the same time, (b) combines related RUs into RU chains, (c) defines the limits of transactions by using the bracket protocol, (d) controls the connection of requests and responses in accordance with control modes specified when the session is started, (e) creates sequence numbers, and (f) associates requests with responses.

data group. In AS/400 Business Graphics Utility, a collection of values that identify the comparisons in a chart. For example, the relative size of the slices in a pie chart or the relative height of the bars in a bar chart. See also *paired data*. Contrast with *data value*.

data integrity. (1) The condition that exists as long as accidental or intentional destruction, alteration, or loss of data does not occur. (2) Within the scope of a unit of work, either all changes to the database management systems are completed or none of them are. The set of change operations are considered an integral set.

data link. The physical connection (communications lines, modems, controller, work stations, other communications equipment), and the rules (protocols) for sending and receiving data between two or more locations in a data network.

data link connection identifier (DLCI). The field in a Q.922 frame that is used for frame relay routing. Each DLCI identifies a frame relay virtual circuit.

data management. The part of the operating system that controls the storing and accessing of data to or from an application program. The data can be on internal storage (for example, database), on external media (diskette, tape, or printer), or on another system.

data mode. In data communications, a time during which BSC is sending or receiving characters on the communications line.

data network identification code (DNIC). A 4-digit code that specifies the X.21 call format used by a network in its International Data Number to call another station. The first three numbers are the data country code, and the last number is the country network identifier.

data queue. An object that is used to communicate and store data used by several programs in a job or between jobs. The system-recognized identifier is *DTAQ.

data stream. All information (data and control commands) sent over a data link usually in a single read or write operation.

data terminal equipment (DTE). That part of a data link that sends data, receives data, and provides the data communications control function according to protocols.

database file. One of several types of the system object type *FILE kept in the system that contains descriptions of how input data is to be presented to a program from internal storage and how output data is to be presented to internal storage from a program. See also *physical file* and *logical file*.

Dataphone digital service (DDS)**. The AT&T line service that allows the customer to transmit data on the line in a digital format.

DBCS. See *double-byte character set (DBCS)*.

DBCS conversion. A function of the operating system that allows a DBCS display station user to enter alphanumeric data and request that the alphanumeric data be converted to double-byte data.

DBCS conversion dictionary. A collection of alphanumeric entries with the double-byte entries that correspond to the alphanumeric entries. It is used by the DBCS conversion function. The system-recognized identifier for the object type is *IGCDCT.

DBCS font table. A system-supplied table that holds either 24x24 or 32x32 pel character images of a double-byte character set. A Japanese 24x24 DBCS font table holds Japanese extended Kanji and user-defined characters. A Korean 24x24 DBCS font table holds a subset of Hanja and user-defined characters. A Traditional Chinese 24x24 DBCS font table holds a subset of primary Traditional Chinese, all secondary Chinese, and user-defined characters. A Simplified Chinese 24x24 DBCS font table holds IBM-supplied Simplified Chinese characters as well as user-defined characters. A 32x32 DBCS font table holds 32x32 pel character images of a double-byte character set, including its user-defined characters. The system-recognized identifier for the object type is *IGCTBL.

DBCS-only field. A field that must contain only double-byte characters.

DCC. See *data country code (DCC)*.

DCE. See *data circuit-terminating equipment (DCE)*.

DDM. See *distributed data management (DDM)*.

DDM file. A system object with type *FILE, created by a user on the local (source) system, that identifies a data file that is kept on a remote (target) system. The DDM file provides the information needed for a local system to locate a remote system and to access the data in the remote data file.

DDS. See *data description specifications (DDS)*.

deallocate. To release a resource that is assigned to a specific task.

debug. To detect, diagnose, and eliminate errors in programs.

debug mode. An environment in which programs can be tested.

debugging mode. (1) A mode in which a program provides detailed output about its activities to aid a user in detecting and correcting errors in the program itself or in the configuration of the program or system. (2) An environment in which programs can be tested.

decompression. A function that exchanges control characters for actual data. See also *compression*.

dedicated save operation. An operation that the user runs to save objects when no other jobs are running. Contrast with *save-while-active operation*.

dedicated service tools (DST). The part of the service function used to service the system when the operating system is not working.

dedicated system. A system intentionally reserved for a single job or task.

default. (1) A value that is automatically supplied or assumed by the system or program when no value is specified by the user. (2) In DDS, the value specified by the user with the DFT or DFTVAL keyword in DDS.

default delivery. The method of delivering messages to a message queue in which messages are placed on the queue without interrupting the job, and the system-assigned reply is sent for any messages requiring a reply.

default focal point. In SNA, a network node that receives alerts from nodes that do not have defined focal points.

default network message queue. A message queue to which messages related to network activity are sent when either the user profile does not have a message queue specified or the message queue named in the user profile cannot be used.

default network output queue. An output queue to which spooled files are sent when either the user does not have an output queue specified or the output queue name in the user profile cannot be used.

default program. A user-specified program that is assumed when no other program is specifically named on a debug

command, or a user-defined program for handling error messages.

default record. A record that consists entirely of default values (numeric fields are filled with zeros; character fields are filled with blanks; and fields of either data type (numeric or character) can be filled with a value specified by the user with the DFT keyword in DDS).

default reply. A system-assigned reply to an inquiry or notify message, which is used when the message queue at which the message arrives is in default delivery mode.

default user name. A system-provided name for a user identification for a computer system that does not want to require separate user identifications.

Defense Data Network (DDN). The MILNET, ARPANET, and TCP/IP networks and protocols.

delayed maintenance. A method of logging changes to an access path for database files and applying the changes the next time the file is opened instead of rebuilding the access path completely or maintaining it immediately.

delete authority. A data authority that allows the user to remove entries from an object; for example, delete messages from a message queue or delete records from a file.

descending key. The values by which data is arranged from the highest value to the lowest value of the key field, in accordance with the rules for comparing data items.

descending sequence. The arrangement of data in order from the highest value to the lowest value, according to the rules for comparing data.

destination service access point (DSAP). In SNA and TCP/IP, a logical address that allows a system to route data from a remote device to the appropriate communications support.

detail record. A record that contains the daily activities or transactions of a business. For example, the items on a customer order are typically stored in detail records. Contrast with *header record*.

DEVD. See *device description*.

device address. A unique identifier for each device so it is recognized by the system.

device class. The generic name for a group of device types. For example, all display stations belong to the same device class.

device configuration. The physical placement of display stations, printers, and so forth; and the configuration descriptions that describe the physical configuration to the system and describe how the configuration will be used by the system.

device description. An object that contains information describing a particular device or logical unit that is attached to the system. The system-recognized identifier for the object type is *DEVD.

device emulation. The programming that allows one device to appear to the user or to a system as another device.

device file. One of several types of the system object type *FILE. A device file contains a description of how data is to be presented to a program from a device or how data is to be presented to the device from the program. Devices can be display stations, printers, diskette units, tape units, or remote systems.

device table. A list of finance devices supported by the AS/400 system to be used by a finance job.

device type. The generic name for a group of devices. For example, 5219 for IBM 5219 Printers.

DFU. See *data file utility (DFU)*.

directory. (1) In the hierarchical file system, a grouping of related files and directories, such as a folder containing related documents. A directory may hold zero or more entries, which refer to other directories and files. (2) In PC Support/400, a list of the files that are stored on a disk or diskette. A directory also contains information about the file, such as size and date of last change.

disconnect (DISC). In communications, the transmission control character that is part of the sequence for disconnecting a switched line.

disconnect (DISC) character. In data communications, the part of the BSC transmission control sequence for ending the connection on a switched line.

disconnected mode (DM). In communications, a response from a secondary station indicating that it is logically disconnected from the link.

disk unit. A physical enclosure containing one or more disk drives.

diskette file. A device file created by the user for a diskette unit.

diskette unit. A physical enclosure containing one or more diskette drives.

diskette writer. A system program that writes spooled files to a diskette unit.

display file. A device file to support a display station.

display image. In 3270 emulation, the x-character block (where x is the maximum number of characters that can fit on the display screen, or 1920 for printers) that contains data in the sequence in which it would appear on the display screen or the printer. When creating the display, the user

can specify the display image with or without field definitions, such as position, length, and other attributes.

display station pass-through. A communications function that allows a user to sign on to one system (either an AS/400 system, System/38, or System/36) from another system (either an AS/400 system, System/38, or System/36) and use that system's programs and data. Sometimes called pass-through.

distributed data management (DDM). The architecture used by the distributed file management (DFM**) and the Distributed Relational Database Architecture (DRDA) protocol to define the protocol used for communicating between two systems using the DFM or the DRDA protocol.

distributed host command facility (DHCF). A function of the operating system that supports the data link between a System/370 terminal using an AS/400 application in an HCF (Host Command Facility) environment.

distributed systems node executive (DSNX). A function of the operating system that receives and analyzes requests from the NetView Distribution Manager licensed program on a host system. If the request is directed to the system that receives it, the request is processed on that system or on a personal computer directly attached to that system. If the request is intended for a different system, it is routed toward its destination.

distribution document. An internal document that contains the document content and the document details for a distribution, such as a note or document.

distribution list. A list of system distribution directory entries, which allows users to send messages, notes, and documents to a group of users in one step.

distribution queue. In SNADS, a list of documents or mail waiting to be sent to users or libraries on remote systems.

distribution service level. In SNADS, the combination of priority, capacity, and protection requirements that must be satisfied to receive or send a distribution. SNADS has service levels of fast, status, data high, and data low. Items with a service level of fast, status, or data high are put on the priority queue. Items with a service level of data low are put on the normal queue.

distribution services. The support provided by the operating system to receive, forward, and send electronic mail in an SNA network.

DLCI. See *data link connection identifier (DLCI)*.

DLO. See *document library object (DLO)*.

DM. See *disconnected mode (DM)*.

document. Any collection of data stored in a document object. All documents and folders on a single AS/400 system make up the document library. A document can contain any type of data stored in it by an application. For example, the SAA OfficeVision/400 application can store notes, memos, reports, and other items; the PC Support/400 shared folders application can store any data that could otherwise be stored in a PC file; an AS/400 application can store any data into a document by using CL commands, such as FILDOC and RPLDOC. The system-recognized identifier for the document object type is *DOC.

document authority. The definition of what actions a user can perform on a document.

document class. A user-defined character string, 1 through 16 characters long, that characterizes a document. It can be used to search for a filed document. For example, a document that is a memo could have a document class of MEMO; a document that is a report, REPORT.

document description. The 1- through 44-character description of a document, assigned by the user when creating or filing the document.

document details. Data that describes the characteristics of a document. For example, the details can include document type, subject, author, and date created.

Document Interchange Architecture (DIA). The rules and structure for the exchange of information between office applications. Document Interchange Architecture includes document library services and document distribution services.

document interchange session. The environment that allows office system users and PC Support/400 users to request document library and distribution services from the host system.

document library. The entire collection of documents and folders on a system.

document library object (DLO). Any system object that resides in the document library, such as RFT and FFT documents, folders, and PC files.

document name. The 1- through 12-character name for documents in folders, assigned by the user when creating the document.

document object name. The 10-character name of a document assigned by the system when a user files the document.

document password. In SAA OfficeVision/400, a string of characters that give authority to use or read a personal document.

document print queue. In SAA OfficeVision/400, a list of output in the output queue or job queue waiting to be printed by the system.

document type. The Document Interchange Architecture type.

double precision. The specification that causes a floating-point value to be stored (internally) in the long format (two computer words). Double precision is known as *long precision* in BASIC.

double-byte character. An entity that requires two character bytes.

double-byte character set (DBCS). A set of characters in which each character is represented by 2 bytes. Languages such as Japanese, Chinese, and Korean, which contain more symbols than can be represented by 256 code points, require double-byte character sets. Because each character requires 2 bytes, the typing, displaying, and printing of DBCS characters requires hardware and programs that support DBCS. Four double-byte character sets are supported by the system: Japanese, Korean, Simplified Chinese, and Traditional Chinese.

download. To send programming instructions and files from a host system to an attached device or to another system. For example, transmitting a type style over a communications line to a 6670 printer.

DST. See *dedicated service tools (DST)*.

DTE. See *data terminal equipment (DTE)*.

dump. (1) In problem analysis and resolution, to write, at a particular instant, all or part of the contents of main or auxiliary storage onto another data medium for the purpose of protecting the data or collecting error information. (2) To copy data from main or auxiliary storage onto an external medium, such as tape, diskette, or printer. (3) Data copied in a readable format from main or auxiliary storage to an external medium such as tape, diskette, or printer.

duplex. (1) Pertaining to communications in which data can be sent and received at the same time. Contrast with *half-duplex*. (2) In AFP support, pertaining to printing on both sides of a sheet of paper.

duplicate key value. The occurrence of the same value in a key field or in a composite key in more than one record in a file.

dynamic. Pertaining to events occurring at run time, or during processing.

dynamic bind. When SQL statements are entered interactively, binding is done automatically (that is, as the SQL statements are entered). See also *bind* and *automatic bind*.

dynamic program call. A connection to another program made during run time of the calling program. A dynamic program call is the only way that an unbound program can connect to another unbound program.

dynamic select/omit. Selection and omission of logical file records performed during processing, instead of when the access path (if any) is maintained. Dynamic select/omit may also be used when no keyed access path exists.

EBCDIC. See *extended binary-coded decimal interchange code (EBCDIC)*.

edit code. A letter or number indicating that editing should be done according to a defined pattern before a field is displayed or printed.

edit description. A description of a user-defined edit code. The system-recognized identifier is *EDTD.

edit word. A user-defined word with a specific format that indicates how editing should be done.

EIA-232. In data communications, a specification of the Electronic Industries Association (EIA) that defines the interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE) using serial binary data interchange.

either field. A field that can contain either double-byte data or alphanumeric data.

electronic customer support. A part of the operating system that allows a customer to access: the question-and-answer (Q & A) function; problem analysis, reporting, and management; IBM product information; and technical information exchange.

electronic overlay. An AFP resource object that is a collection of predefined data, such as lines, shading, text, boxes, or logos, that can be merged with variable data on a page while printing. The system-recognized identifier for the object type is *OVL.

element. In a list of parameter values, one value.

element mark. In Advanced Function Printing Utilities/400, a mark that is used to show the position of an element on a display; for example, '*B005' where B means bar code and 005 is the fifth element.

emulation. Imitation of one system or device by another.

end node. In SNA, a node in an APPN network that can be a source or target node, but does not provide any routing or session services to any other node.

end point. The system that is the origin or destination of a session.

end-of-file delay. An interval during which the system holds a file open after the normal end of the file is reached until one or more records are updated or added to the end of the file. The length of the interval can be specified on the EOFDLY parameter.

end-of-text (ETX) character. The BSC transmission control character used to end a logical set of records that began with the start-of-text character. Contrast with *end-of-transmission-block (ETB) character*.

end-of-transmission (EOT) character. The BSC transmission control character used to end transmission with the remote system.

end-of-transmission-block (ETB) character. The BSC transmission control character used to end a block of records. Contrast with *end-of-text (ETX) character*.

enhanced logical link control (ELLC). An X.25 protocol that allows the transfer of data link control information between two adjoining SNA nodes that are connected through an X.25 packet-switching data network. ELLC enhances error detection and recovery. Contrast with *physical services header (PSH)* and *qualified logical link control (QLLC)*.

entry format. The description of a personal directory entry. Each personal directory entry has an identical structure. The entry structure determines the type and size of each field in a personal directory entry.

error log. A record of machine checks, device errors, and media statistics.

escape message. A message that reports a condition that caused the program to end before the requested function was complete.

Ethernet. A type of local area network that is supported by the Operating System/400 licensed program. OS/400 Ethernet provides support for the Digital Equipment Corporation, Intel Corporation, and Xerox** standard (Ethernet Version 2) and the IEEE 802.3 standard. These local area networks use Carrier Sense Multiple Access with Collision Detection (CSMA/CD) as the media access method.

exclude authority. An object authority that prevents the user from using the object or its contents. Contrast with *all authority*.

| **export**. An external symbol defined in a module or program | that is available for use by other modules or programs. See | also *external symbol*. Contrast with *import*.

extended binary-coded decimal interchange code (EBCDIC). A coded character set consisting of 8-bit coded characters.

| **external data**. Data that persists over the lifetime of an acti- | vation group and maintains last-used values whenever a pro- | cedure is reentered. The scope of external data is that of | the enclosing activation group. All procedures called within | the activation group recognize the external data. Contrast | with *automatic data* and *local data*.

external message queue. The part of the job message queue that sends messages between an interactive job and the work station user. For batch jobs, messages sent to the external message queue appear only in the job log.

external procedure. A procedure that is not contained within a block.

| **external symbol**. An item defined in a high-level language | program that represents such things as procedures or vari- | ables. Resolving external symbols is the means by which the | binder connects modules to form a bound program or a | service program.

externally described file. A file in which the records and fields are described to the system when the file is created, and used by the program when the file is processed.

| **facsimile machine**. A functional unit that converts images | to signals for transmission over a telephone system or that | converts received signals back to images. (T)

| **fax**. (1) The printed copy received from a facsimile | machine. (T) (2) To transmit an image using a telephone | system and facsimile machines. (T) (3) The use of a tele- | phone system for the electronic transmission and receipt of | hard-copy images. (T)

| **fax machine**. Synonym for *facsimile machine*. See *facsimile* | *machine*.

fidelity. In AFP support, the degree of exactness required when processing the input data stream for printing a file. Different levels of fidelity can be specified, which determine how errors are handled (such as substituting fonts when a font named in the data stream cannot be found).

field. A group of related bytes (such as name or amount) that is treated as a unit in a record.

field definition. In IDDU, information that describes the characteristics of data in a field, such as its name, length, and data type. A field definition resides in a data dictionary. d

field description. Information that describes the characteristics of data in a field.

FIFO. See *first-in first-out (FIFO)*.

file definition. In IDDU, information that describes the contents and characteristics of a file. A file definition resides in a data dictionary. See also *field definition* and *record format definition*.

file description. The description of a file and its contents.

file identifier. A 3-character identifier used for files being joined in Query/400 for a query. The identifiers are used during a query definition to uniquely identify each file.

file name. The name used by a program to identify a file. See also *label*.

file overrides. Attributes specified at run time that change the attributes specified in the file description or in the program.

file separator. The pages produced at the beginning of each output file and used to separate the file from the other files being sent to an output device.

file system. In the hierarchical file system, the underlying system support that manages I/O operations to files and controls the format of information on the storage media. A file system allows applications to create and manage files on storage devices and to perform I/O operations to those files.

FILE type. A data type that allows the program to read input and write output in AS/400 Pascal.

finance communications. The data communications support that allows programs on an AS/400 system to communicate with programs on finance controllers, using the SNA LU session type 0 protocol.

finance device. A device, such as the 4700 Finance Communications System devices and the 3694 Document Processor, that performs functions specifically related to the finance industry. The 3180, 3270, and 5250 work stations are not finance devices.

first-character forms control (FCFC). A method that specifies the format of printed output. The first character of each record determines the format.

first-in first-out (FIFO). (1) A queuing technique in which the next request to be processed from a queue is the request of highest priority that has been on the queue for the longest time. (2) In REXX, a queuing technique in which the next item to be retrieved is the item that has been on the queue for the longest time. Contrast with *last-in first-out (LIFO)*.

first-level folder. A folder name that is not preceded by another folder name. A first-level folder is the first folder name in a folder path. For example, if folder A is a first-level folder, folder path A/B indicates that folder B is within folder A, and that folder A is within the root folder. See also *folder path*.

flag. (1) The bit sequence 01111110 used to mark a frame in SDLC. (2) Information about the extended attribute that is stored with the extended attribute.

floating currency symbol. A currency symbol that appears immediately to the left of the far left position in an edited field. Contrast with *fixed currency symbol*.

floating-point. A method of encoding real numbers within the limits of finite precision available on computers.

floating-point constant. (1) A number shown as an optional sign followed by one or more digits and a decimal point, which may be at the end. (2) A numeric constant shown as an optional sign, followed by the letter D or E, followed by a 1- to 3-digit integer constant. For example, 3E-02, which is 3 times 10 to the -2 power or 0.03.

floating-point format. (1) In binary floating-point representation, the storage format that represents a binary floating-point value.

flow. The passing of a message from one process to another. The passing of messages of a particular type between processes. For example, Distributed Relational Database Architecture (DRDA) flows are those that consist only of messages described by the DRDA protocol as part of the DRDA protocols.

flow control. In OSI, procedures that control the amount of data than can be sent from one node to another. Flow control is used to prevent a node from sending data to another node faster than the receiver can handle it.

focal point. An APPN network node that is the destination of alerts. A focal point allows a customer to centrally manage a network.

fold. To continue data on the next line. Contrast with *truncate*.

folder. A directory for documents. A folder is used to group related documents and to find documents by name. The system-recognized identifier for the object type is *FLR. See also *document library object*. Compare with *library*.

folder path. A folder name, followed by one or more additional folder names, where each preceding folder is found. For example, path A/B indicates that folder B is within folder A, and that folder A is in the root folder. See also *subfolder*.

folderless document. A document in the document library that is not in any folder.

font. (1) An assortment of characters of a given size and type style. (2) A particular style of type (for example, Bodini or Times Roman) that contains definitions of character sets, marker sets, and pattern sets.

font ID. A number that identifies the character style and size for certain printers.

font resource. A resource object on the AS/400 system that is required to print AFPDS documents on a printer. The three types of font resources are coded fonts, character sets, and code pages. The system-recognized identifier for the object type is *FNTRSC.

footer. In SAA OfficeVision/400, one or more lines of text that prints at the bottom of every page of a document, such as a page number, the date, an outline heading, or the document ID.

form. (1) In AFP support, a physical sheet of paper on which data is printed. Synonymous with medium, physical page, and sheet. (2) In query management, an object that describes how to format the data for printing or displaying a report.

form definition. An AFP resource object that defines the characteristics of the printed media; for example, overlays to be used, text suppression, position of page data on the form, and number and modifications of a page. The system-recognized identifier for the object type is *FORMDF.

form feed. An ASCII printer control, X'0C', that causes the printer to eject the current page. All jobs printing on a page printer should end with a form feed, which forces the last page to print.

form type. A 10-character identifier, assigned by the user, that identifies each type of form used for printed output.

format. (1) A defined arrangement of such things as characters, fields, and lines, usually used for displays, printouts, files, or documents. (2) The arrangement or layout of fields in a record. (3) The arrangement or layout of data on a storage medium, such as disk, tape, or diskette. (4) To set the block size for the 9332 Disk Unit, either automatically by the system or specifically by the user. (5) In BASIC, a representation of the correct form of a command or statement. (6) To arrange information on a page, in a file, or on a display screen. (7) In PC Support/400, to prepare a diskette so that it can be used by a personal computer.

format selector. A user-defined program (either a CL or a high-level language program) that determines where a record should be placed in the database when an application program does not pass a record format name for a record being added to a logical file.

Formatted Data Object Content Architecture (FD:OCA). A defined collection of constructs used to interchange formatted data.

frame. In communications, the unit of transmission sent and received by the data link layer, one of the seven layers defined in the ISO standard.

frame relay. A protocol for routing frames through the network based on the address field (data link connection identifier) in the frame and for managing the route or virtual connection.

full duplex. Synonym for *duplex*.

function check. A notification that an unexpected condition has stopped the running of a program.

function key. A keyboard key that allows the user to select keyboard functions or programmer functions. Contrast with *character key*.

gateway. A program used to connect two systems that use different communications protocols.

GDDM. See *graphical data display manager (GDDM)*.

general-purpose library. The library shipped with the system that contains IBM-provided objects required for many system functions and user-created objects that are not explicitly placed in a different library when they are created. Named QGPL.

generate. In CSP/AD, to produce a representation of an application program, map group, or table that can be interpreted by Cross System Product/Application Execution.

generic. Relating to, or characteristic of, a whole group or class.

generic alert. A product-independent method of encoding alert data by means of both (a) code points indexing short units of stored text and (b) text.

generic name. (1) The characters common to object names that can be used to identify a group of objects. A generic name ends with an asterisk (*). For example, ORD* identifies all objects whose names begin with the characters ORD. (2) In the hierarchical file system, a path name that contains one or more wildcard characters.

generic search. In SAA OfficeVision/400, a type of search that searches all documents or personal directories authorized to a user for information that corresponds to a search value that contains a string of characters followed by an asterisk. The asterisk indicates to the system that the user wants to identify all instances of the specified character string.

global variable. A named entity within query management that can be assigned a value used for communications between an application program and Query Management/400.

graph. In Performance Tools/400, the displayed, printed, or plotted output that represents the horizontal and vertical axis variables specified by the user for a collection of performance data.

graph format. In Performance Tools/400, a template used to display performance and historical graphs. The graph format consists of such things as titles, axis variables, and the type of graph.

graphic character set. A set of graphic characters in a code page.

graphical data display manager (GDDM). A function of the operating system that processes both text and graphics for output on a display, printer, or plotter. Contrast with *presentation graphics routines (PGR)*.

graphical user interface. A type of user interface that takes advantage of high-resolution graphics. A graphical user interface includes a combination of graphics, the object-action relationship, the use of pointing devices, menu bars and other menus, overlapping windows, and icons.

graphics. (1) Pictures and illustrations. (2) Pertaining to charts, tables, and their creation.

graphics data format (GDF) file. A picture definition in a coded order format used internally by GDDM and, optionally, providing the user with a lower-level programming interface than the GDDM application programming interface.

graphics symbol set. In GDDM, an object that can contain either lines or images. The system-recognized identifier for the object type is *GSS. See also *vector symbol set (VSS)* and *image symbol set (ISS)*.

group address. In communications, a multidestination address associated with one or more stations on a given network. Contrast with *individual address*.

group authority. Authority to use objects, resources, or functions from a group profile.

group data area. A data area that is automatically created when an interactive job becomes a group job. This data area is shared by all jobs in the group but cannot be used by jobs outside the group.

group job. One of up to sixteen interactive jobs that are associated in a group with the same work station device and user.

group job name. The name that identifies a given job within a group.

group message queue. A message queue associated with a group of jobs. When the message queue is set to break or notify mode in the active group job, it is set to the same mode in any job in the group that is transferred to or to any job that gains control when the active group job is canceled.

group profile. A user profile that provides the same authority to a group of users.

half-duplex. Pertaining to data communications that can be sent in only one direction at a time. Contrast with *duplex*.

half-session. In SNA, one of the locations in a logical connection in a network. See also *session*.

handle. In application programming interfaces, a variable that represents an object.

hard-copy reference. In SAA OfficeVision/400, a description of printed mail that is kept on the system with electronic mail. This allows you to keep track of both types of mail using OfficeVision/400.

hardware. Physical equipment, rather than programs, procedures, rules, and associated information.

header label. A special set of information on a diskette or tape that describes the contents of the diskette or tape.

header record. A record that contains information, such as customer name and customer address, that is common to detail records.

help module. In user interface manager, the smallest part of a panel group object that can be separately displayed. A help module can be used for contextual help, extended help, an Infoseeker topic, or a hypertext information node.

hex. See *hexadecimal*.

hexadecimal. Pertaining to a numbering system with a base of 16.

HFS. See *hierarchical file system (HFS)*.

hierarchical file system (HFS). A part of the operating system that includes the application programming interfaces and the underlying file system support. HFS enables an application written in a high-level language to create, store, retrieve, and manipulate data on a storage device. The view of the data to the end user is a hierarchical directory structure similar to IBM DOS.

hierarchy. In the InfoSeeker function, the organization of topics within a search index.

high-level data link control (HDLC). A form of communications line control that uses a specified series of bits rather than control characters to control data transmission over a communications line.

high-level language (HLL). A programming language, such as RPG, BASIC, PL/I, Pascal, COBOL, and C used to write computer programs.

high-level language (HLL) pointer. A source pointer that the programmer declares in the user program.

history log. A summary of the system activities, such as system and job information, device status, system operator messages, and a record of program temporary fix (PTF) activity on the system. The history log is identified by the name QHST, and the system-recognized identifier for the object type is *MSGQ.

HLL. See *high-level language (HLL)*.

hop. The transmission from one location to the next in a network.

hops. The number of systems that a distribution passes through to its destination.

host. The controlling or highest-level system in a data communications configuration; for example, an AS/400 system is the host system for the work stations connected to it.

Host Command Facility (HCF). A feature available on a System/370, 43xx, and 30xx host system that enables a user on the host system to use applications on an AS/400 system or other systems as if they were using remotely attached 5250-type display stations.

host command processor (HCP). The SNA logical unit of the programmable store system store controller.

host print transform. An Operating System/400 print function that converts an SNA character string (SCS) data stream into an ASCII data stream. The ASCII data stream is then formatted and sent to an ASCII printer through one or more hardware connections, such as PC Support/400, 3477, or 3487 work stations. This single location of the transform allows for consistent ASCII printing through any of the hardware connections.

I/O. See *input/output*.

I/O processor. See *input/output processor (IOP)*.

IBM Advanced Data Communications for Stores (ADCS). The IBM licensed program that functions on the System/370 host processor for host system to point-of-sale system communications.

IBM Operating System/400 Version 2 (OS/400). Pertaining to the IBM licensed program that can be used as the operating system for the AS/400 system.

IBM PC Support/400 Version 2. The IBM licensed program that provides system functions to an attached personal computer.

IBM SAA OfficeVision/400 Version 2. The IBM licensed program that allows users to prepare, send, and receive mail; schedule items on calendars; maintain directories of names and addresses; file and retrieve documents; and create and maintain distribution lists. SAA OfficeVision/400 also provides word processing functions and the capability to work on behalf of other users.

ICF. See *intersystem communications function (ICF)*.

ICF file. A device file that allows a program on one system to communicate with a program on another system. There can be one or more sessions with the same or different communications devices at the same time.

IDDU. See *interactive data definition utility (IDDU)*.

identifier. (1) The name of something. (2) A sequence of bits or characters that identifies a user, program, device, or system to another user, program, device, or system. (3) In the C language, a sequence of letters, digits, and underscores used to identify a data object or function. (4) In COBOL, a data name that is unique or is made unique by the correct combination of qualifiers, subscripts, or indexes. (5) In FORTRAN, a lexical unit that names a language object; for example, the names of variables, arrays, and program units. The name of a declared unit. (6) In Pascal, a lexical unit that names a language object; for example, the names of variables, arrays, records, labels, and procedures. The name of a declared item. (7) In PL/I, a single alphabetic character or a string of alphabetic characters, digits, and break characters that starts with an alphabetic character. (8) For SQL, see *delimited identifier* and *ordinary identifier*.

IDLC. See *ISDN data link control (IDLC)*.

IDP. See *interchange document profile (IDP)*.

IEEE. Institute of Electrical and Electronics Engineers.

IGC. Abbreviation used in commands and keywords to represent double-byte character set functions.

ILE. See *Integrated Language Environment (ILE)*.

image. An electronic representation of an original document recorded by a scanning device.

image symbol set (ISS). In GDDM, a graphics symbol set in which each character is treated as a small image and is described by a rectangular array of display points. Characters in an image symbol set are always drawn in a fixed size. Contrast with *vector symbol set*; see also *graphics symbol set*.

immediate maintenance. A method of maintaining keyed access paths for database files. This method updates the access path whenever changes are made to the database file associated with the access path. Contrast with *rebuild maintenance* and *delayed maintenance*.

immediate message. A message that is created when it is sent. Contrast with *predefined message*.

implicit. Capable of being understood from something else, though unexpressed.

import. (1) In the SAA Application Development Manager/400 product, to copy AS/400 objects and source members from an AS/400 library to the SAA Application Development Manager/400 environment. (2) A reference to an external symbol defined in another module or program. Contrast with *export*.

IMS/VS. See *Information Management System for Virtual Storage (IMS/VS)*.

include. In SAA OfficeVision/400, to insert text into a document when the document is printed.

independent work station. See *programmable work station (PWS)*.

index name. In COBOL, a user-defined word that names an index.

index search. See *InfoSeeker*.

indication. In OSI, a service primitive issued by a service provider to call a procedure by a service user.

indicator. (1) A 1-character or 2-character code that is used by a program to test a field or record or to tell when certain operations are to be performed. (2) An internal switch used by a program to remember when a certain event occurs and what to do when that event occurs. (3) In the SAA COBOL/400 licensed program, a 2-character code that is used by a program to test a field or record or to tell when certain operations are to be performed. (4) In the SAA

RPG/400 licensed program, a 2-character code that is used as a logical variable or statement label.

indirect user. In SAA OfficeVision/400, a person enrolled in the system distribution directory who receives mail but never signs on to view it. An indirect user receives printed mail only. Contrast with *direct user*.

individual address. In communications, an address associated with a particular station on the network. Contrast with *group address*.

infinity. A name referring to an indefinitely great number.

information display. A display that presents information to a user, such as the status of the system, but that rarely requests a response.

information element. In ISDN, the messages that are exchanged over the D-channel between the system and ISDN. For example, when a call is set up, a message is sent to the network containing several information elements, one of which is the number of the remote system. Other information elements may be present.

information frame (I-frame). In communications, a transmission frame that is sequentially numbered and used to transmit data.

Information Management System for Virtual Storage (IMS/VS). A general purpose system that enhances the capabilities of OS/VS for batch processing and telecommunication. It allows users to access a computer-maintained database through remote terminals.

informational message. A message that provides information to the user about the system as compared with a completion message, which indicates success, and escape or diagnostic messages, which indicate failure.

InfoSeeker. An information retrieval function of the online system information that provides how-to and explanatory topics through search indexes that supplement the online help. The topics can be browsed through a hierarchical organization or searched directly.

initial program. (1) A user-profile program that runs when the user signs on and after the command processor program QCMD is started. QCMD calls the first program. (2) In COBOL, a program that is placed into an initial state every time the program is called in a run unit.

initial program load (IPL). The process that loads the system programs from the system auxiliary storage, checks the system hardware, and prepares the system for user operations.

initialize. To set the addresses, switches, or the contents of storage to zero, or to the starting value set by the manufacturer.

initiator. In OSI Communications Subsystem/400, the application entity that starts an application association. Contrast with *responder*.

inline. Spooled input data that is read into a job by a reader.

inline data file. A file created by a Data (// ATA) command that is included as part of a job when the job is read from an input device or a database file. The file is deleted when the job ends.

input field. A field specified in a display file or database file that is reserved for information supplied by a user. Contrast with *output field*.

input file. (1) In COBOL, a file from which data is read while the program is running. (2) In RPG, a database or device file that has been opened to allow records to be read. Contrast with *output file*.

input stream. (1) A group of records submitted as a batch job that contains CL commands for one or more jobs and data from one or more inline data files. (2) In RJE, data sent to the host system.

input/output. Data provided to the computer or data resulting from computer processing.

input/output adapter (IOA). (1) A functional unit or a part of an I/O controller that connects devices to an I/O processor. (2) For devices, the electrical circuits on a logic card that connect one device to another.

input/output processor (IOP). A functional unit or the part of an I/O controller that processes programmed instructions and controls one or more input/output devices or adapters.

inquiry message. A message that gives information and requests a reply.

inquiry program. (1) A program that allows an operator to get information from a disk file. (2) A program that runs while the system is in inquiry mode.

instruction. (1) In the C language, a statement that specifies an operation to be performed by a computer. (2) In COBOL and Pascal, one or more clauses, the first of which starts with a keyword that identifies the instruction. Instructions affect the flow of control, provide services to the programmer, or both. (3) In REXX, one or more clauses that describe some course of action to be taken by the language processor. Instructions may be assignments, keyword instructions, or commands.

integer. (1) A positive or negative whole number. (2) An SQL data type indicating that the data is a binary number with a precision of 31 bits. (3) In COBOL, a numeric constant or a numeric data item that does not include any digit position to the right of the assumed decimal point.

Integrated Language Environment (ILE). Pertaining to a set of constructs and interfaces that provides a common run-time environment and run-time bindable application program interfaces (APIs) for all ILE-conforming high-level languages.

integrated services digital network (ISDN). A CCITT Recommendation that defines an interface to a network that can carry voice, data, and image over the same communications line. See also *basic rate interface (BRI)* and *primary rate interface (PRI)*.

integrity. See *data integrity*.

Intelligent Printer Data Stream* (IPDS). (1) An all-points-addressable data stream that allows users to position text, images, and graphics at any defined point on a printed page. (2) In GDDM, a structured-field data stream for managing and controlling printer processes, allowing both data and controls to be sent to the printer.

interactive. Pertaining to the dialog-like exchange of information between people and a computer.

interactive data definition utility (IDDU). A function of the operating system that can be used to externally define the characteristics of data and the contents of files.

interactive job. A job started for a person who signs on to a work station.

interactive mode. In query management, the query mode associated with a query instance that allows users to interact with the query commands while a procedure is running.

interactive subsystem. A subsystem in which interactive jobs are processed.

interactive terminal facility (ITF). An asynchronous communications function that allows an AS/400 system to communicate with applications that can send and receive data, such as electronic mail, memos, library members, and data files.

interchange document profile (IDP). The Document Interchange Architecture object that contains information associated with each document. For example, the interchange document profile can contain authors, keywords, dates, and so on. The interchange document profile is one of many model objects that DIA has defined to keep information about the document. A profile consists of a set of subprofiles.

interface. A shared boundary. An interface might be the hardware to connect two devices or it might be a part of main storage, or registers used by two or more computer programs.

intermediate block check. In BSC, a check that verifies each record, rather than the contents of the total block, when large blocks of data are received. See also *intermediate-text-block (ITB) character*.

intermediate-text-block (ITB) character. The BSC transmission control character used to divide a block of text into smaller groups of text for an intermediate block check. See also *intermediate block check*.

internal data. In COBOL, the data described in a program excluding all external data items and external file connectors. Items described in the Linkage Section of a program are treated as internal data.

internal storage. All main and auxiliary storage in the system.

international standard. A standards document that is given final approval by the International Organization for Standardization.

interpreter. A program that translates and runs each instruction of a high-level programming language before it translates and runs the next instruction.

interrecord-separator character (IRS). In BSC, a transmission control character that is used to separate records within a block of data.

intersystem communications function (ICF). A function of the operating system that allows a program to communicate interactively with another program or system.

intrinsic function. A function supplied by the AS/400 BASIC licensed program.

inventory. The quantity of materials or goods on hand.

inverse. Opposite in order, nature, or effect.

IOP. See *input/output processor (IOP)*.

IPDS. See *Intelligent Printer Data Stream (IPDS)*.

IPL. See *initial program load (IPL)*.

ISDN. See *integrated services digital network (ISDN)*.

ISDN data link control (IDLC). An asynchronous, balanced data link protocol used between two systems to exchange information over an ISDN B-channel.

ISO. International Organization for Standardization.

Japanese dictionary. See *DBCS conversion dictionary*.

job. (1) A unit of work to be done by a computer. (2) The highest-level logical entity of the Integrated Language Environment (ILE) model. A job is a collection of resources and data, and consists of one or more activation groups. See also *activation group*. (3) In the SAA OfficeVision/400 calendar function, an item that schedules a control language (CL) command to run at any date and time.

job accounting. A system function that collects information about a job's use of system resources and records that information in a journal.

job action. The network attribute that controls the handling of a job submitted from remote locations through either the SNADS network or RSCS.

job control authority. A special authority that allows a user to: change, delete, display, hold, and release all files on output queues; hold, release, and clear job queues and output queues; start writers to output queues; hold, release, change, and end other users' jobs; change the class attributes of a job; end subsystems; and start (do an IPL of) the system.

job date. The date associated with a job. The job date usually assumes the system date, but it can be changed by the user.

job description. A system object that defines how a job is to be processed. The object name is *JOBD.

job log. A record of requests submitted to the system by a job, the messages related to the requests, and the actions performed by the system on the job. The job log is maintained by the system program.

job message queue. A message queue that is created for each job. A job message queue receives requests to be processed (such as commands) and sends messages that result from processing the requests. A job message queue consists of an external message queue and a set of program message queues.

job name. The name of the job as identified to the system. For an interactive job, the job is assigned the name of the work station at which the job was started; for a batch job, the name is specified in the command used to submit the job. Contrast with *qualified job name*.

job queue. An object that contains a list of batch jobs waiting to be processed by the system. The system-recognized identifier for the object type is *JOBQ.

job schedule. An object that contains entries for jobs to be submitted at a specified time and date. These job schedule entries can also be used to schedule recurring jobs. The system-recognized identifier for the object type is *JOBSCD.

job schedule entry. An entry in the job schedule object that describes the job to be submitted. The user can specify attributes of the job and when the job will be submitted.

join field. A comparison field that identifies records from two files to be combined into one record.

join logical file. A logical file that combines (in one record format) fields from two or more physical files.

journal. A system object that identifies the objects being journaled, the current journal receiver, and all the journal receivers on the system for the journal. The system-recognized identifier for the object type is *JRN. See also *journal receiver*.

journal code. A 1-character code in a journal entry that identifies the category of the journal entry. For example, F identifies an operation on a file; R identifies an operation on a record, and so forth.

journal entry. A record in a journal receiver that contains information about database files.

journal entry identifier. The fields in a journal entry that identifies the journal code, the journal entry type, the date and time of the entry, the job name, the user name, and the program name.

journal entry qualifier. The part of a journal entry that identifies the name of the object for which the journal entry was created.

journal entry type. A 2-character field in a journal entry that identifies the type of operation of a system-generated journal entry or the type of journal entry of a user-generated journal entry; for example, PT is the entry type for a write operation.

journal receiver. A system object that contains journal entries added when changes are made to an object, for example, when an update is made to a file being journaled. The object type is *JRNRCV. See also *journal*.

Julian date. A date format that contains the year in positions 1 and 2, and the day in positions 3 through 5. The day is represented as 1 through 366, right-adjusted, with zeros in the unused high-order positions. For example, the Julian date for April 6, 1987 is 87096.

key. The value used to identify a record in a keyed sequence file.

key field. A field used to arrange the records of a particular type within a file member.

keyboard shift. In DDS, a characteristic that can be specified for a field in a display file that automatically shifts the display station keyboard to control what the display station user can enter into the field. In IDDU and DDS, the keyboard shift can also be specified in database files, but only applies when these fields are referred to in a display file.

keyboard type. The physical key arrangement and assignments for the keyboard shipped from the factory.

keyed sequence. An order in which records are retrieved that is based on the contents of key fields in records.

keyed sequence access path. An access path to a database file that is arranged according to the contents of key fields contained in the individual records.

keyword. (1) A mnemonic (abbreviation) that identifies a parameter in a command. (2) In the SAA OfficeVision/400 program, a user-defined word used as one of the search values to identify a document during a search operation. (3) In DDS, a name that identifies a function. (4) In REXX,

| a symbol reserved for use by the language processor in a certain context. Keywords include the names of the instructions and ELSE, END, OTHERWISE, THEN, and WHEN. (5) A name that identifies a parameter used in an SQL statement or SQL precompiler command. See also *parameter*.

keyword functions. The result of processing DDS keywords in a record format specified on an operation.

label. (1) The name of a file on a diskette or tape. (2) An identifier of a command or program statement generally used for branching. (3) In REXX, a clause that consists of a single symbol followed by a colon.

LAN. See *local area network (LAN)*.

| **language ID**. See *language identifier (language ID)*.

| **language identifier (language ID)**. The 3-character representation that identifies the cultural preference for language-related processing and is associated with an object, such as a document. For example, the language identifier is used by text search services to determine how to process the text of a document.

last-in first-out (LIFO). In REXX, a queuing technique in which the next item to be retrieved is the item most recently placed in the queue.

layer. (1) In SNA, a grouping of related functions that are logically separate from the functions in other groups; the functions in one layer can be changed without affecting functions in the other layers. See also *data flow control layer*, *path control layer*, and *transmission control layer*. (2) In a network architecture, a group of services, functions, and protocols that is complete from a conceptual point of view, that is one out of a set of hierarchically arranged groups, and that extends across all systems that conform to the network architecture. (T) See also *application layer*, *data link layer*, *network layer*, *physical layer*, *presentation layer*, *session layer*, and *transport layer*.

| **Lempel-Ziv (LZ)**. A technique for compressing data. This technique replaces some character strings, which occur repeatedly within the data, with codes. The encoded character stings are then kept in a common dictionary, which is created as the data is being sent.

level checking. A function that compares the record level identifiers of a file to be opened with the file description that is part of a compiled program to determine if the record format for the file changed since the program was compiled.

library. A system object that serves as a directory to other objects. A library groups related objects, and allows the user to find objects by name. The system-recognized identifier for the object type is *LIB.

library list. A list that indicates which libraries are to be searched and the order in which they are to be searched. The system-recognized identifier is *LIBL.

library name. A user-defined word that names a library.

library-assigned document name (LADN). A unique name, which includes a time stamp and a system name, that is assigned by a system in the office network to a document when it is filed in the document library. On the AS/400 system, the time-stamp part of the library-assigned document name is included in a 10-character name that becomes the document object name. d

licensed internal code. The layered architecture below the machine interface (MI) and above the machine, consisting of the model-independent and the model-unique licensed internal code. The licensed internal code is a proprietary system design that carries out many functions, including, but not limited to storage management, pointers and addressing, program management functions, exception and event management, data functions, I/O managers and security.

licensed internal code fix. A temporary solution to, or bypass of, a defect in a current release of the licensed internal code.

licensed program. A separately orderable program, supplied by IBM, that performs functions related to processing user data. Examples of licensed programs are PC Support/400, SAA COBOL/400, AS/400 Application Development Tools, SAA OfficeVision/400, and so on.

LIFO. See *last-in first-out (LIFO)*.

line. The physical path in data transmission.

line configuration. The process of creating configuration descriptions for the lines that make up a data processing system. See also *controller configuration* and *device configuration*.

line description. An object that contains information describing a particular communications line that is attached to the system. The system-recognized identifier for the object type is *LIND.

line number. The number that precedes a line of information in a printout or on a display. This number can be up to 5 digits long, from 00001 through 99999.

line printer. A device that prints a line of characters as a unit. Contrast with *page printer*.

lines per inch (lpi). The number of characters that can be printed vertically within an inch.

link. (1) In hypertext, an author-defined association between two information nodes. (2) In IDDU, to connect a database file on disk with a file definition in a data dictionary. Contrast with *unlink*. (3) In SNA, the combination of the link connection (the transmission medium) and two link stations (one at each end of the link connection). See also *link level*.

link level. (1) In SNA, the combination of the transmission connection, protocol, devices, and programming joining network nodes. (2) A part of Recommendation X.25 that defines the link protocol used to get data into and out of the network across the duplex line connecting the subscriber's equipment to the network.

Link Problem Determination Aid-2 (LPDA-2). A second version of the LPDA command set. In addition to all the functions of LPDA-1, LPDA-2 also supports modem configuration, dial command, and an open and close contact command.

link protocol. The rules for sending and receiving data at the link level.

list ID. A two-part name by which a distribution list is known. The two-part name allows distributions to be sent to both local and remote systems.

| **LMI**. See *local management interface (LMI)*.

local. Pertaining to a device or system that is connected directly to your system or a file that is read directly from your system, without the use of a communications line. Contrast with *remote*.

local address. In SNA, an address used in a peripheral node in place of a network address and transformed to or from a network address by the boundary function in a subarea node.

local area network (LAN). The physical connection that allows the transfer of information among devices located on the same premises.

local controller. A functional unit within the system that controls the operation of one or more directly attached input/output devices or communications lines. Contrast with *remote controller*.

local data. Data that is recognized only by the procedure that defines it. The scope of local data is that of the enclosing activation group. Contrast with *automatic data* and *external data*.

local data area. A 1024-byte data area that can be used to pass information between programs in a job. A separate local data area is automatically created for each job.

local location address. In SNA, the address of the logical unit.

local location name. The name by which your system is known to other systems in an SNA network. Equivalent to an SNA local logical unit name. Contrast with *remote location name*.

| **local management interface (LMI)**. The interface between
| the frame-relay data terminal equipment (DTE) and the frame
| handler, which provides the status and configuration information about the permanent virtual circuits (PVCs) available at
| the frame relay network.

local network address. In OSI, a network address that identifies the local node. See also *network address*.

local node. In the OSI Communications Subsystem/400 licensed program, the node from which one views the rest of the OSI network—the node for which resources are defined.

local system. For interactive jobs, the system to which the display device is directly attached. For batch jobs, the system on which the job is being processed.

local work station. A work station that is connected directly to the system without a need or data transmission functions.

lock. The process by which integrity of data is ensured by preventing more than one user from accessing or changing the same data or object at the same time.

lock state. A condition defined for an object that determines how it is locked, how it is used (read or write), and whether the object can be shared (used by more than one job).

logarithm. The exponent that indicates the power to which a number is raised to produce a given number.

logic. The systematized interconnection of digital switching functions, circuits, or devices.

logical channel. In a packet-switching data network, a path over which data flows between the network and the sending or receiving data terminal equipment.

logical expression. An expression consisting of logical operators and/or relational operators that can be evaluated to a value of either true or false.

logical file. A description of how data is to be presented to or received from a program. This type of database file contains no data, but it defines record formats for one or more physical files. d

logical file member. A named logical grouping of data records from one or more physical file members.

logical link control (LLC). A protocol for data-link-level transmission control. The protocol was developed by the IEEE 802 committee, and is common to all LAN standards.

logical link control (LLC) protocol. In a local area network, the protocol that governs the assembling of transmission frames and their exchange between data stations independently of the medium access control protocol. For X.25 LLCs, see *enhanced logical link control (ELLC)*, *qualified logical link control (QLLC)*, and *physical services header (PSH)*.

logical record. In COBOL, the most inclusive data item. The level number for a logical record is 01.

logical resource. In OSI, an abstract resource—such as a layer entity. Contrast with *physical resource*.

logical unit (LU). In SNA, one of three types of network addressable units that serve as a port through which a user accesses the communications network.

logical unit (LU) 6.2. A type of logical unit that supports general communications between programs in a distributed processing environment. LU 6.2 is characterized by (a) a peer relationship between session partners, (b) efficient use of a session for multiple transactions, (c) comprehensive end-to-end error processing, and (d) a generic application program interface (API) consisting of structured verbs that are mapped into a product implementation. Synonym for *advanced program-to-program communications (APPC)*.

long comment. Up to a full-screen description of a field, record format, or file. Long comments are typed when the field, record format, or file is created or changed, and displayed either from IDDU or Query.

long format. In binary floating-point storage formats, the 64-bit representation of a binary floating-point number, not-a-number, or infinity. Contrast with *short format*.

long precision. In BASIC, the level of precision in which values printed in fixed-point format have a maximum of 14 significant digits, and values printed in floating-point format have a maximum of 15 significant digits.

low-entry networking (LEN) node. A node in an APPN network that uses the LU session type 6.2 node type 2.1 architecture without the APPN extension.

LPDA-2. See *Link Problem Determination Aid-2 (LPDA-2)*.

lpi. See *lines per inch (lpi)*.

LU. See *logical unit (LU)*.

LZ. See *Lempel-Ziv (LZ)*.

machine interface (MI). The interface, or boundary, between the operating system and the licensed internal code.

macro. In REXX, a program that performs certain operations, such as text editor operations, in applications.

macroinstruction. A single instruction that represents a set of instructions.

mail log. In SAA OfficeVision/400, a record of all the electronic and printed mail that an office user has sent or received.

main storage. All addressable storage where programs are run. Synonymous with *memory*. See also *control storage*.

main storage pool. A division of main storage, which allows the user to reserve main storage for processing a job or group of jobs, or to use the pools defined by the system.

major/activity token. In OSI, the session-layer token that controls activities and major synchronize operations.

management. In OSI, a synonym for *systems management*.

management domain. (1) In OSI, a synonym for *network management domain*. (2) In OSI X.400, a set of one or more message transfer agents and zero or more user agents that comprise a system capable of handling messages and is managed by either an administration or private company.

management services. In SNA, one of the types of network services in control points and physical units. Management services are the services provided to assist in the management of SNA networks, such as problem management, performance and accounting management, configuration management, and change management.

manager. In the OSI Communications Subsystem/400 licensed program, a synonym for *managing process*.

managing process. In OSI, the part of a systems management application that monitors and controls the resources of an agent process. In OSI Communications Subsystem/400, the managing process can send operator commands to—and receive event reports from—its agent processes.

manual answer. In data communications, a line type that requires operator actions to receive a call over a switched line.

manual call. In data communications, a line type requiring operator actions to place a call over a switched line.

Manufacturing Automation Protocol (MAP). In OSI, a specification developed by industrial users to provide a common set of protocols to allow communications between computers and factory floor equipment in the manufacturing environment. It is based on a subset of the open systems interconnection (OSI) standard.

MAP. See *Manufacturing Automation Protocol (MAP)*.

map group. In CSP/AD, a collection of maps that have the same type of device characteristics. Maps in a map group can be shared between CSP/AE applications. The system-recognized identifier is *CSPMAP.

mapped conversation. In advanced program-to-program communications (APPC), a temporary connection between an application program and an APPC session in which the system provides all the information on how the data is formatted

mapping. A representation of one thing to another.

mapping table. An object that contains a set of hexadecimal characters used to map data from one character set and code page to another.

marker. (1) In GDDM, a symbol centered on a point. Line charts may use markers to indicate the plotted points. (2) In

hardware, reflective material placed on magnetic tape to indicate the beginning or ending of the recording area.

mask. A pattern of characters that is used to control the keeping, deleting, or testing of portions of another pattern of characters.

master file. A collection of permanent information, such as a file of customer addresses.

matrix. An arrangement in rows and columns.

medium. The disk, tape, or diskette used to store information in a save or restore operation.

megabyte. A unit of measure for storage capacity. For main storage, 1 megabyte equals 1 048 576 bytes (1024 x 1024); for auxiliary storage (disk, diskette, and tape), 1 megabyte equals 1 000 000 bytes (1000 x 1000).

member. Different sets of data, each with the same format, within one database file. See also *source member*.

memory. In PC Support/400, program-addressable storage from which instructions and other data can be loaded directly into registers for subsequent processing.

menu. (1) A displayed list of items from which a user can make a selection. The system-recognized identifier for the object type is *MENU.

message. A communication sent from a person or program to another person or program.

message description. Information describing a particular message.

message file. An object that contains message descriptions. The system-recognized identifier for the object type is *MSGF.

message identifier. A seven-character code that identifies a predefined message, and is used to get the message description from a message file.

message line. An area on the display where messages are displayed.

message queue. A list on which messages are placed when they are sent to a user ID or device description. The system-recognized identifier for the object type is *MSGQ.

message reference key. A key assigned to every message on a message waiting line. This key is used to remove a message from a message waiting line, to receive a message, and to reply to a message.

migration. The process of moving data and source from one computer system to another without converting the data.

mnemonic. A symbol or abbreviation chosen to help the user remember the significance or meaning of the symbol. For example, CRTUSRPRF is a mnemonic for the Create User Profile command.

mode. The session limits and common characteristics of the sessions associated with advanced-program-to-program communications (APPC) devices managed as a unit with a remote location.

mode description. A system object created for advanced-program-to-program communications (APPC) devices that describes the session limits and the characteristics of the session, such as the maximum number of sessions allowed, maximum number of conversations allowed, the pacing value for incoming and outgoing request or response units, and other controlling information for the session. The system-recognized identifier for the object type is *MODD.

model file. In performance, a complete representation of a system. It includes both the system configuration and the set of workloads running on the configuration.

modem (modulator/demodulator). A device that converts data from the computer to a signal that can be sent over a communications line (modulator), and converts the communications signal received to data for the computer (demodulator).

modification level. A distribution of all temporary fixes issued since the previous modification level. A change in modification level does not add new functions or change the programming support category of the release to which it applies. A new release is shipped with a modification level of 0. When the release is shipped with the service changes incorporated, the modification level is incremented by 1. See also *release* and *version*.

module. (1) An independent unit that is part of a total structure. (2) In online education, a unit of instruction that is part of an education course. notation. (3) In the Integrated Language Environment (ILE) model, the object that results from compiling source code. A module cannot be run. To be run, a module must be bound into a program. See also *bind*. Contrast with *program*.

multiple device file (MDF). A device file in which the maximum number of program devices is greater than one.

Multiple Virtual Storage (MVS). An alternative name for OS/VS2. See also *operating system* and *virtual storage (VS)*.

multipoint. In data communications, pertaining to a network that allows two or more stations to communicate with a single system on one line.

multipoint line. A line or circuit connecting several stations.

name server. In TCP/IP, a server application that associates domain names with internet addresses. Usually, all name servers are arranged in a tree structure corresponding to the domain naming hierarchy, and at each domain one or more machines will assume this naming task.

NAUN. See *nearest active upstream neighbor (NAUN)*.

NDM. See *normal disconnected mode (NDM)*.

nearest active upstream neighbor (NAUN). In the IBM Token-Ring Network, the station sending data directly to another station in the ring.

NetView. (1) Pertaining to an IBM licensed program used to monitor a network, manage it, and diagnose its problems. (2) In OSI, pertaining to an IBM licensed program that is used to monitor a network, manage it, and diagnose its problems. The NetView licensed program can be used to provide network management services for OSI Communications Subsystem/400.

network. A collection of data processing products connected by communications lines for exchanging information between stations.

network address. In OSI, an address that identifies a particular node. A network address can consist of (a) a network entity title only, (b) an NSAP address only, or, (c) both a network entity title and an NSAP address.

network architecture. The logical structure and operating principles of a computer network. (T) The operating principles of a network include those of services, functions, and protocols.

network entity title. In OSI, a title that identifies the network entity on a given node. Because a node can have only one network entity, the network entity title uniquely identifies a given node. Network entity titles are represented in the same format as NSAP addresses.

network file. In object distribution, a file (either a physical file or a save file) sent by one user to one or more other users. A network file is placed on the recipient's message queue when it arrives at the destination system.

network ID. In TCP/IP, that part of the internet address that defines the network. The length of the network ID depends on the type of network class (A, B, or C).

network interface (NWI). The physical interface that allows a user to connect to the integrated services digital network (ISDN).

network interface description. An AS/400 communications object that represents the physical interface to the ISDN. The network interface description must be configured in addition to the line, controller, and device descriptions. The system-recognized identifier for the object type is *NWID.

network job. In object distribution, a batch input stream sent by one user to one or more users in the network as defined in the system distribution directory.

network job entry. In object distribution, an entry in the network job table that specifies the system action required for incoming network jobs sent by a particular user or group of users. Each entry is identified by the user ID of the originating user or group.

network job table. In object distribution, a table containing entries that control the system action required for incoming network jobs.

network layer. In OSI architecture, the layer that provides services to establish a path between open systems with a predictable quality of service. (T)

network management. The process of planning, organizing, and controlling a communications-oriented system.

network management domain. In OSI, a manager and the agents that it manages. An agent can participate in more than one network management domain. In OSI Communications Subsystem/400, the agent at a local node is always part of the management domain of the manager at that local node.

network message. In object distribution, a message sent by one user to one or more users enrolled in the system distribution directory with the Send Network Message (SNDNETMSG) command.

network mode. In OSI, a synonym for *network QOS mode*.

network node. A node that can define the paths or routes, control route selection, and handle directory services for APPN.

network node server. A network node that is directly connected to an end node or a low-entry networking end node, and has been assigned to service the end node session requests.

network QOS mode. In the OSI Communications Subsystem/400 licensed program, a set of X.25 connection-mode quality-of-service (QOS) values that determine the type of connection established between two nodes. Synonymous with *network mode*.

network termination (NT). In ISDN, equipment that provides the function necessary for the operation of the access protocols by the network. See also *network termination 1 (NT1)*, *network termination 2 (NT2)*, *terminal equipment 1 (TE1)*, and *terminal equipment 2 (TE2)*.

network termination 1 (NT1). In ISDN, an end point for the network's transmission line. Network termination 1 is responsible for the physical layer characteristics (of the OSI reference model), such as ending the line transmission, monitoring performance, and timing.

network termination 2 (NT2). In ISDN, an end point for the network's transmission line. Network termination 2 is responsible for the network layer, the data link layer, and the remaining functions of the physical layer (not included in network termination 1) of the OSI reference model. Examples include communications controllers and public branch exchanges (PBXs).d

network user identification (NUI). In X.25, network-specific information that is used by the network to uniquely identify the data terminal equipment (DTE) originating a switched virtual call.

network-layer service access point (NSAP). In OSI, a service access point in the network layer. (I)

next record. The record that logically follows the current record of a file.

next system. A node in the SNADS network that is physically connected to the local system, and through which distribution items can be routed.

node. (1) One of the systems or devices in a network. (2) A location in a communications network that provides host-processing services. (3) For APPN, see *network node* and *end node*. (4) In hypertext, an information unit containing information about a single topic and linked to one or more other nodes. (5) In X.25, a point where packets are received, stored, and forwarded to another location (or data terminal equipment) according to a routing method defined for the network.

nonadjacent destination node. In the OSI Communications Subsystem/400 licensed program, a destination node that is connected to a different subnetwork from the local node. To communicate with a nonadjacent destination node requires the use of a relay node.

nonlabeled tape. A tape that has no labels. Tape marks are used to indicate the end of the volume and the end of each data file.

nonpaired data. In AS/400 Business Graphics Utility and GDDM, data that is specified such that each X-value has a set of Y-values associated with it.

nonprogrammable work station (NWS). A work station that does not have processing capability and does not allow the user to change its functions. Contrast with *programmable work station (PWS)*.

nonswitched line. A connection between computers or devices that does not have to be made by dialing.

nontext document data. In SAA OfficeVision/400, data to be used for charts or illustrations stored in a document.

normal disconnected mode (NDM). A nonoperational mode of an unbalanced data link in which the secondary station is logically disconnected from the data link and, therefore, cannot transmit or receive information.

normal queue. In SNADS, a queue that contains distribution entries with a service level of data low. Contrast with *priority queue*.

normal response mode (NRM). An operational mode of an unbalanced data link in which the secondary station starts transmission only as the result of receiving explicit permission, by polling, from the primary station.

notify delivery. The method of delivering messages to a message queue in which the work station user is notified that a message arrived. The signal is a light or an audible alarm.

notify message. A message that describes a condition for which a program requires a reply from the calling program, or for which a reply is automatically sent to the program.

notify object. A message queue, a data area, or a database file that contains information identifying the last successful commitment operation. This information can be used by the programmer to find a restarting point for an application following an abnormal end to the system or routing step processing.

NRZI. Non-return-to-zero (inverted) recording.

NSAP. See *network-layer service access point (NSAP)*.

NSAP address. In OSI, an address that identifies a service access point in the network layer. NSAP addresses must be unique within the OSI network where they are used. NSAP addresses are assigned by naming authorities.

NUI. See *network user identification (NUI)*.

null. (1) The name for an EBCDIC character that represents hex 00. See *null character*. (2) In SQL, a special value that indicates the absence of information.

null character. The character hex 00 used to represent the absence of a displayed or printed character.

null record. In binary synchronous communications, a record that contains no data, only the data link control characters STX ETX.

null string. A character or bit string with a length of zero.

null value. A parameter position for which no value is specified.

number. In REXX, a character string consisting of one or more decimal digits optionally preceded by a plus or minus sign, and optionally including a single period that represents a decimal point. A number can also have a power of 10 suffix in conventional exponential notation: an E (uppercase or lowercase) followed optionally by a plus or minus sign then followed by one or more decimal digits defining the power of 10.

numeric field. An area that is reserved for a particular unit of information and that can contain only the digits 0 through 9

numeric variable. The name of a numeric data item whose value is assigned or changed during program processing.

NWI. See *network interface (NWI)*.

object. (1) A named storage space that consists of a set of characteristics that describe itself and, in some cases, data. An object is anything that exists in and occupies space in storage and on which operations can be performed. Some examples of objects are programs, files, libraries, and folders. (2) A visual part of the interface that the user can work with to perform a task. Icons and text are examples of objects.

object auditing. A function of the OS/400 operating system that creates audit records for specified types of access to an object.

object authority. A specific authority that controls what a system user can do with an entire object. For example, object authority includes deleting, moving, or renaming an object. There are three types of object authorities: object operational, object management, and object existence.

object description. The characteristics (such as name, type, and owner name) that describe an object.

object existence authority. An object authority that allows the user to delete the object, free storage of the object, save and restore the object, transfer ownership of the object, and create an object that was named by an authority holder.

object management authority. An object authority that allows the user to specify the authority for the object, move or rename the object, and add members to database files.

object name. The name of an object. Contrast with *qualified name*.

object operational authority. An object authority that allows the user to look at the description of an object and use the object as determined by the user's data authorities to the object.

object owner. A user who creates an object or to whom the ownership of an object was reassigned. The object owner has complete control over the object.

object type. In query management, the substring following the query command name that specifies the type of query object to be processed.

observability. The property of an object, which is derived from data stored with the object, that allows source to be retrieved from the object, allows the object to be re-created without being recompiled, and allows the object to be symbolically debugged.

ODP. See *open data path (ODP)*.

OfficeVision*. See *IBM SAA OfficeVision/400 Version 2*.

offline. Pertaining to the operation of a functional unit that is not under the continual control of the system. Contrast with *online*.

offset. (1) The distance from the beginning of an object to the beginning of a particular field, or for substring operations, the number of character positions from the beginning of a field. (2) In SAA OfficeVision/400, the first page from a printed copy from the 6670 printer that sticks out from the remaining pages to separate one job from another. (3) In GDDM, the number of character grid units from a reference point.

online. Pertaining to the operation of a functional unit that is under the continual control of the system.

open data path (ODP). A control block created when a file is opened. An ODP contains information about the merged file attributes and information returned by input or output operations. The ODP only exists while the file is open.

open field. A field that can contain a mixture of alphanumeric and double-byte characters with shift-out and shift-in characters marking the transitions. See also *either field* and *only field*.

operating system. A collection of system programs that control the overall operation of a computer system.

Operational Assistant. A part of the operating system that provides a set of menus and displays for end users to do commonly performed tasks, such as working with printer output, messages, and batch jobs.

OPM. See *original program model (OPM)*.

optimization level. The level of efficiency for processing a program, which is determined by the application programmer. When the code is optimized on the system, the system uses processing shortcuts to reduce the amount of system resources necessary to produce the same output. The processing shortcuts are then translated by the system into machine code, which allows the program to run more efficiently.

original program model (OPM). The set of functions for compiling source code and creating high-level language programs on the AS/400 system before the Integrated Language Environment (ILE) model was introduced.

originator/recipient name (O/R name). In OSI X.400, the name of the user (the originator and recipient of messages) and other attributes.

OS/400. See *IBM Operating System/400 Version 2 (OS/400)*.

OS/400 Cross System Product/Application Execution (OS/400 CSP/AE). A function of the operating system that gives the user the capability to run CSP/AE applications, which have been defined and generated in one of the Cross System Product/Application Development (CSP/AD) environments.

output field. A field specified in a display file, database file, printer file, or ICF file that is reserved for the information processed by a program.

output queue. An object that contains a list of spooled files to be written to an output device, such as a printer or a diskette. The system-recognized identifier for the object type is *OUTQ.

overflow. The condition that occurs when the last line specified as the overflow line to be printed on a page has been passed.

overlay. To write over (and therefore destroy) an existing file.

override. (1) To specify attributes at run time that change the attributes specified in the file description or in the program. (2) The attributes specified at run time that change the attributes specified in the file description or in the program.

overrun. The loss of data because a receiving device is unable to accept data at the rate it is transmitted.

owner. The user who creates an object (or is named the owner of an object).

pacing. In SNA, a technique by which the receiving system controls the rate of transmission of the sending system to prevent overrun.

package. In a distributed relational database, the control structure produced when the SQL statements in an application program are bound to a relational database management system (DBMS). The DBMS uses the package to process SQL statements encountered during the processing of a statement.

packed decimal format. Representation of a decimal value in which each byte within a field represents two numeric digits except the far right byte, which contains one digit in bits 0 through 3 and the sign in bits 4 through 7. For all other bytes, bits 0 through 3 represent one digit; bits 4 through 7 represent one digit. For example, the decimal value +123 is represented as 0001 0010 0011 1111.

packet. (1) In data communications, a sequence of binary digits, including data and control signals, that is transmitted and switched as a composite whole. (I) (2) In X.25, a data transmission information unit. A group of data and control characters, transferred as a unit, determined by the process of transmission. Commonly used data field lengths in packets are 128 or 256 bytes.

packet assembler/disassembler (PAD). A functional unit that enables data terminal equipment (DTE) not equipped for packet switching to use a packet-switched network.

packet window. A specified number of packets that can be sent by the DTE before it receives an acknowledgement from the receiving station.

PAD. See *packet assembler/disassembler (PAD)*.

page. (1) A unit of storage equal to 512 bytes. (2) A 512-byte block of information that can be moved between auxiliary storage and main storage. (3) Each group of records in a subfile that are displayed at the same time. (4) One printer form. (5) In GDDM, the picture or chart. All specified graphics are added to the current page. An output statement always sends the current page to the device. (6) To move information up or down on the display.

page definition. An AFP resource that contains the formatting controls for line data. A page definition can include controls for the number of lines per logical page, font selection, print direction, and mapping individual fields to positions on the logical page. The system-recognized identifier for the object type is *PAGDFN.

page printer. In AFP support, any of a class of printers that accepts composed pages, constructed of composed text and images, among other things.

page segment. An AFP resource object that can contain text and images and can be positioned on any addressable point on a page or an electronic overlay. The system-recognized identifier for the object type is *PAGSEG.

paired data. In AS/400 Business Graphics Utility and GDDM, data that is specified so that every X value has only one Y value associated with it. See also *data group*. Contrast with *nonpaired data*.

panel. In UIM, a visual presentation of data on the screen.

panel group. An object that contains a collection of any of the following: display formats, print formats, or help information. The system-recognized identifier for the object type is *PNLGRP.

parameter. (1) A value supplied to a command or program that is used either as input or to control the actions of the command or program. (2) In REXX, information entered with a command name to define the data on which a command processor operates and to control the execution of the command.

parameter list. A list of values that provide a means of associating addressability of data defined in a called program with data in the calling program. It contains parameter names and the order in which they are to be associated in the calling and called program.

parity. The state of being either even-numbered or odd-numbered. A parity bit is a binary number added to a group of binary numbers to make the sum of that group either always odd (odd parity) or always even (even parity).

parity bit. A binary digit added to a group of binary digits to make the sum of all the digits, including the added binary digit, either odd or even as preestablished. (T)

party. In telephony, an addressable end point of a telephone call. See also *virtual party*.

PASA. See *program automatic storage area (PASA)*.

pass-through. See *display station pass-through*.

path. (1) In a network, any route between any two nodes. (T) (2) For PC Support/400 using DOS, the sequence of directories, specified by the user, to search when looking for a program or data. (3) For PC Support/400 using the OS/2* licensed program, the route used to locate files on a disk or diskette, consisting of a drive and directories. (4) In SNA, the set of data links, data link control layers, and path control layers that a path information unit travels through when sent from the transmission control layer of one half-session to the transmission control layer of another half-session.

path control layer. In SNA, the layer that routes all messages to data links and half-sessions.

path control network. In SNA, the part of the network that includes the data link control and path control layers.

pattern. In REXX, the parts of a parsing template that allow a string to be split by the explicit matching of strings (literal patterns) or by the specification of numeric positions (positional patterns). Parentheses may be supplied to create a variable pattern, a pattern whose value is derived from a variable.

PC file. A file stored on a personal computer.

PC Support. See *IBM PC Support/400 Version 2*.

peer. A general term for the corresponding node or entity with which one communicates.

peer-to-peer networking. See *Advanced Peer-to-Peer Networking (APPN)*.

pending. Pertaining to a request that was submitted and that is awaiting processing.

performance. That part of the system that is evident in elapsed time. Performance is largely determined by three factors: throughput, response time, and availability.

performance monitor. A function of the operating system that observes system and device activity, and records these observations in a database file.

permanent error. In the token-ring network manager, an error—for example, a hardware component failure—that can be corrected only by external intervention.

permanent objects. Objects, such as database files or programs, that stay in the system until a user with the required authority deletes them.

permanent virtual circuit (PVC). A virtual circuit that has a logical channel permanently assigned to it at each data terminal equipment (DTE). A call establishment protocol is not required. The permanent virtual circuit establishes the identity of the called party within the network services contract. Contrast with *switched virtual circuit (SVC)*.

physical file. A description of how data is to be presented to or received from a program and how data is actually stored in the database. A physical file contains one record format and one or more members.

physical file member. A named subset of the data records in a physical file.

physical services header (PSH). An X.25 protocol used by IBM Systems Network Architecture (SNA) data terminal equipment (DTE). Physical services header provides address services for physically connected systems or devices. The AS/400 system does not support PSH.

physical unit (PU). In SNA, one of three types of network addressable units. A physical unit exists in each node of an SNA network to manage and monitor the resources (such as attached links and adjacent link stations) of a node, as requested by a system services control point logical unit (SSCP-LU) session.

physical unit type. In SNA, the classification of a physical unit according to the type of node in which it resides. The physical unit type is the same as its node type; that is, a type 1 physical unit resides in a type 1 node, and so on.

pitch. The number of characters printed per inch.

plotter. In AS/400 Business Graphics Utility, a device for drawing a chart on paper or transparencies.

pointer. The symbol shown on a display or window that a user can move with a pointing device, such as a mouse.

point-of-sale. In retail communications and Point-of-Sale Communications Utility/400, pertaining to a method of providing information to support sales and of collecting the resulting sales information from retail devices located in stores.

point-to-point. Pertaining to data transmission between two locations without the use of any intermediate display station or computer.

point-to-point line. A communications line that connects a single remote station to a computer. Contrast with *multipoint line*.

poll. To determine if any remote device on a communications line is ready to send data.

polling. The process whereby a controlling station contacts the attached devices to avoid contention, to determine operational status, or to determine readiness to send or receive data.

pool. A division of main or auxiliary storage.

port. (1) System hardware where the I/O devices are attached. (2) An access point (for example, a logical unit) for data entry or exit. (3) A functional unit of a node through which data can enter or leave a data network. (4) In data communications, that part of a data processor that is dedicated to a single data channel for the purpose of receiving data from or transmitting data to one or more external, remote devices.

post. (1) To add information in a record to keep that record current. (2) To note the occurrence of an event.

Post Telephone and Telegraph Administration (PTT). An organization, usually a government department, that provides data communication services in countries other than the USA and Canada. Examples of PTTs are the Bundespost in Germany and the Nippon Telephone and Telegraph Public Corporation in Japan.

power down. An AS/400 command to turn the power off and bring an orderly end to system operation.

precision. See *single precision* and *double precision*.

predefined message. A message whose description is created and stored in a message file before it is sent by the program.

predefined value. A fixed value defined by IBM that has a special use in the control language and is reserved in the operating system. A predefined value usually has an asterisk (*) as the first character in the value.

presentation graphics routines (PGR). A group of routines within the operating system that allows business charts to be defined and displayed procedurally through function routines. Contrast with *graphical data display manager (GDDM)*.

prestart job. A job that starts running before the remote program sends a program start request.

previous release. The last required release of the system (such as Release 1.0) prior to the current release (such as Release 2.0), including any modification levels (such as Release 1.0 Modification Level 1 or Modification Level 2) that were not required.

previous system. The system that sent the TELNET or pass-through request that brought the user to the current system.

primary file. (1) In the DDS for a join logical file, the first physical file specified on the JFILE keyword. Contrast with *secondary file*. (2) For certain types of join operations using Query/400, the first of all files that are joined in a query definition. The data from this file is used in every record formed by a join specification.

primary focal point. A network node that receives alerts from nodes that the user has defined in a sphere of control. Contrast with *default focal point*.

primary language. The national language installed on the system as the default language used to display and print information. The primary language is also used to service the system. Contrast with *secondary language*.

primary rate interface (PRI). In ISDN, an interface that provides 23 (or 30 in Europe) 64 000 bps data channels (B-channels) and one 64 000 bps signaling channel (D-channel). Also known as *23/30B + D*.

primary system name. In SNADS, the system name of an AS/400 system.

print descriptor. An object used to manage printing that is created and maintained by PrintManager/400. Print descriptors for printing describe where a print job will be printed, how a print job will be processed, and how output will appear. The print descriptors contain capabilities and defaults of options used for printing. The system-recognized identifier for the object type is *PDG.

print descriptor group. An object used to store print descriptors so they can be managed effectively on a system. The system-recognized identifier for the object type is *PDG.

print options. Specifications for printing a document.

print text. An option that allows the user to specify a line of text at the bottom of a list.

printer file. A device file that determines what attributes printed output will have. A particular printer may or may not support all of the attributes specified in a printer file.

printer ID. The identification code assigned to printers.

printer writer. A system program that writes spooled files to a printer. See also *diskette writer* and *spooling writer*.

priority queue. In SNADS, a queue that contains distribution queue entries for distributions with a service level of fast, status, or data high. When send times and queue depths are satisfied for both the priority and normal queues, the priority queue is serviced first. Contrast with *normal queue*.

private authority. The authority specifically given to a user for an object that overrides any other authorities, such as the authority of a user's group profile or an authorization list.

private management domain (PRMD). In OSI X.400, a private company or noncommercial organization that handles a management domain.

PRM. See *program resolution monitor (PRM)*.

problem analysis. The process of finding the cause of a problem. For example, a program error, device error, or user error.

problem log. A record of problems and of the status of the analysis of those problems.

procedure call. A call made to a procedure within a module in a bound program. Contrast with *program call*.

process. In SAA SystemView System Manager/400, a combination of systems management applications that accomplishes one or more customer tasks or a part of a task. A process may contain other processes.

process access group (PAG). A group of job-related objects that may be paged in and out of storage in a single operation when a job (process) enters or leaves a long wait.

processing. The action of performing operations and calculations on data.

processing unit. The part of the system that performs instructions and contains main storage.

processor. (1) A device for processing data from programmed instructions. It may be part of another unit. (2) One or more integrated circuits that process coded instructions and perform a task.

production library. A library containing objects needed for normal processing. Contrast with *test library*.

product load. (1) In SAA SystemView System Manager/400, an object that contains the control information about an option. The object type is *PRODLOD. A product load is identified by the product identifier (PRDID), release (RLS), option (OPTION), and load identifier (LODID) parameters. (2) The smallest logical collection of objects that can make a product option.

profile. Data that describes the characteristics of a user, program, device, or remote location.

program automatic storage area (PASA). A system object that contains call level information for each program on the program stack. The PASA can also contain space for program variables, which is allocated when the program object is called.

program call. A call made to a bound or unbound program. See also *dynamic program call* and *static program call*. Contrast with *procedure call*.

program device. A symbolic device that a program uses instead of a real device (identified by the device name). When the program uses a program device, the system redirects the operation to the appropriate real device.

program device override. The attributes specified at run time that change the attributes of the program device.

program ID. A 1- to 8-character string entered from a finance device and associated with an AS/400 finance transaction program. Lists of valid program IDs and their associated application programs are maintained in program tables.

program message queue. An object used to hold messages that are sent between program calls of a routing step. The program message queue is part of the job message queue.

program name. A user-defined word that identifies a COBOL source program.

program object. One of two machine object classifications. It includes those objects used in programs that get their definition from an object definition table. Program objects are used as the parameter or values of machine instructions.

program resolution monitor (PRM). A program that translates the intermediate representation of a program into the machine language for use by the computer. The program resolution monitor is used by the programming language compilers to complete the translation of a source program into machine language instructions.

program stack. A list of programs linked together as a result of programs calling other programs with the CALL instruction, or implicitly from some other event, within the same job.

program table. A list of the AS/400 finance applications for use in an AS/400 finance job. Each table entry consists of a program ID and the program name and library associated with that ID. Program IDs received in data streams from finance devices are located in the program table to determine the AS/400 application that should be called.

program temporary fix (PTF). A temporary solution to or bypass of a problem diagnosed by IBM as resulting from a defect in a current unaltered release of a licensed program. Contrast with *licensed internal code fix*.

program variable. A named changeable value that can exist only within programs. Its value cannot be obtained or used when the program that contains it is no longer running.

program-described file. A file for which the fields in the records are described only in the programs that process the file. To the operating system, the record appears as a character string. d

programmable work station (PWS). A work station that has some degree of processing capability and allows the user to change its functions. Contrast with *nonprogrammable work station (NWS)*.

programming interface. The supported method through which customer programs request software services. The programming interface consists of a set of callable services provided with a product.

programming interface for customers. A synonym for *programming interface*.

prompt. A reminder or a displayed request for information or user action. The user must respond to allow the program to proceed.

protocol. A set of rules controlling the communication and transfer of data between two or more devices or systems in a communications network.

PTF. See *program temporary fix*.

PTT. See *Post Telephone and Telegraph Administration (PTT)*.

public authority. The authority given to users who do not have any specific (private) authority to an object, who are not on the authorization list (if one is specified for the object), and whose group profile has no specific authority to the object.

purge. In Performance Tools/400, a job attribute that specifies whether a job is to be marked eligible to be moved out of main storage to auxiliary storage when entering a long wait or leaving the activity level.

PVC. See *permanent virtual circuit (PVC)*.

QCMD. The IBM-supplied control language processor that interprets and processes CL commands for the system.

QGPL. See *general-purpose library*.

QLLC. See *qualified logical link control (QLLC)*.

QSRV. The IBM-supplied user profile for a service representative.

QSYS. See *system library*.

qualified job name. A job name and its associated user name and a system-assigned job number. Contrast with *job name*.

qualified logical link control (QLLC). An X.25 protocol that allows the transfer of data link control information between two adjoining SNA nodes that are connected through an X.25 packet-switching data network. The QLLC provides the qualifier "Q" bit in X.25 data packets to identify packets that carry logical link protocol information. Contrast with *enhanced logical link control (ELLC)* and *physical services header (PSH)*.

qualified name. The name of the library containing the object and the name of the object. Contrast with *object name*.

qualifier. In data processing, all names in a qualified name other than the far right, which is called the simple name.

quality of service (QOS). In OSI, a value that specifies certain performance characteristics of a service, session, or link. In OSI Communications Subsystem/400, quality of service is provided at the network layer.

Query. The shortened name for the Query/400 licensed program.

query definition. In Query/400, information about a query that is stored in the system. The system-recognized identifier for the object type is *QRYDFN.

query management form. In query management, the type name of the OS/400 object on the AS/400 system that is comparable to the term form object as used for the Systems Application Architecture (SAA) solution. The system-recognized identifier for a query management form is *QMFORM.

query management object. In query management, a collective term to describe any of the query management objects: query, form, or procedure.

query management procedure. The name used in Query Management/400 to describe a source physical file member that contains query procedure language statements.

query management query. In query management, the type name of the OS/400 object on the AS/400 system that is comparable to the term query object as used for the Systems Application Architecture (SAA) solution. The system-recognized identifier for a query management query is *QMQRY.

rack. A free-standing framework that holds the devices and card enclosure.

random by key. A processing method for files in which the value in the key field identifies the records to be processed.

random processing. A method of processing in which records can be read from, written to, or deleted from a file order requested by the program that is using them.

read authority. A data authority that allows the user to look at the contents of an entry in an object or to run a program.

read-from-invited-program-devices operation. An input operation that waits for input from any one of the invited program devices for a user-specified time.

read-from-one-program-device operation. An input operation that will not complete until the specified device has responded with input.

rebuild maintenance. A method of maintaining keyed access paths for database files. This method updates the access path only while the file is open, not when the file is closed; the access path is rebuilt when the file is opened.

receiver. See *journal receiver*.

receiver directory. Summary information about the journal receivers that are or were attached to the specified journal and are still known to the system.

record. A group of related data, words, or fields treated as a unit, such as one name, address, and telephone number.

record format. A named part of a file that identifies records of a specified record format description.

record format definition. In IDDU, information that describes the arrangement or layout of fields in a record. A record format definition resides in a data dictionary.

record format description. A description of the characteristics of the fields (for example, type and length) and the arrangement of the fields in a record created by the user.

record separator. In BSC, a control character used to indicate the end of one record and the beginning of another.

record type. The classification of records in a file. Records of the same type have the same fields in the same order. For program-described files, these records have record identification codes; for externally described files, the records have the same record format name.

recoverability. The degree or extent to which the system can be restored to an operational condition after a system failure.

recovery. The process of rebuilding databases after a system failure.

recursion. A programming technique in which a program or routine calls itself to perform each successive step in the solution of a problem and uses the output returned from this call in completing the current step.

recursion level. The position of a program in a program stack. The first occurrence of a program in a job has a recursion level of 1, the second occurrence of the same program has a recursion level of 2, and so on.

reference code. The four-character name of a status or error condition.

relational operator. (1) The reserved words or symbols used to express a relational condition or a relational expression. are: .GT., .GE., .LT., .LE., .EQ., and .NE.. They are defined as greater than, greater than or equal to, less than, less than or equal to, equal to, and not equal to, respectively.

relative record number. A number that specifies the relationship between the location of a record and the beginning of a database file, member, or subfile. For example, the first record in a database file, member, or subfile has a relative record number of 1.

release token. In OSI, the token that controls the orderly release of an association.

remote. Pertaining to a device, system, or file that is connected to another device, system, or file through a communications line.

remote controller. A device or system, attached to a communications line, that controls the operation of one or more remote devices.

remote device. A device whose controller is connected to an AS/400 system by a communications line.

remote job entry (RJE). A function of the AS/400 Communications Utilities licensed program that allows a user to submit a job from a display station on the AS/400 system to a System/370-type host system.

remote location name. Any other system with which your system can communicate in a network. This corresponds to the remote location name specified in the communications configuration.

remote name server. In TCP/IP, the function that allows a system to get an internet address from a remote site rather than from its own host table.

remote system. Any other system in the network with which your system can communicate.

remote work station. A work station that is connected to the system by data communications.

request message. A message that requests a function from the receiving program.

requester. In PC Support/400, a program that requests services from another program (a server). Each PC Support/400 function has a server and a requester.

resolve. In programming, to change a predefined, symbolic value to the actual value of the item being processed. For example, a symbolic value of *LAST defined for the name of a file member is resolved to the name of the last member when the member is processed.

resource. Any part of the system required by a job or task, including main storage, devices, the processing unit, programs, files, libraries, and folders.

resource name. A name assigned by the system to a line, controller, or device that is connected to the system.

response. In SDLC, a frame transmitted by a secondary station. Stations using asynchronous balanced mode send both commands and responses.

restore. To copy data from tape, diskette, or a save file to auxiliary storage. Contrast with *save*.

restricted state. The status in which a user places a system (by ending all subsystems) to do a specific function, such as saving storage, saving the system, or restoring user profiles. Other jobs cannot be active on the system while it is in a restricted state.

REstructured eXtended eXecutor (REXX) language. A general-purpose programming language, particularly suitable for CL commands, or programs for personal computing. Procedures and programs written in this language can be interpreted by the REXX/400 interpreter. See also *REXX/400*.

RESULT. A REXX special variable that is set by the RETURN instruction in a called routine. The RESULT special variable is dropped if the called routine does not return a value.

retail communications. The data communications support that allows programs on an AS/400 system to communicate with programs on point-of-sale systems, using SNA LU session type 0 protocol.

retail pass-through. An OS/400 system program that supports routing of user data between a System/370-type host processor and a retail controller using a single AS/400 system. Both the SNA upline facility and the retail communications support use separate intersystem communications function sessions.

retry. To resend data a prescribed number of times or until the data is received correctly.

return code. For printer files, display files, and ICF files, a value sent by the system to a program to indicate the results of an operation by that program.

REXX/400. The Operating System/400 implementation of the Systems Application Architecture Procedures Language. REXX/400 is a programming language that is supported by an interpreter provided as part of the Operating System/400 licensed program. See also *REstructured eXtended eXecutor (REXX) language*.

right-justify. To print text with an even right margin by adding extra space throughout a line.

RJE. See *remote job entry (RJE)*.

roll back. To remove changes that were made to database files under commitment control since the last commitment boundary.

rollback. (1) The process of restoring data changed by an application program or user to the state at its last commitment boundary.

root folder. The folder on the system that contains all other folders. The system-recognized identifier is *ROOT.

route. The path that network traffic follows from its source to its destination.

router. A part of the PC Support/400 licensed program that handles requests to send and receive data from applications on the personal computer and routes them to the appropriate applications on the AS/400 system.

routing. The list of users who are to receive an item when it is distributed, including all users named specifically and those users named on distribution lists by the sender.

routing entry. An entry in a subsystem description that specifies the program to be called to control a routing step that runs in the subsystem.

routing step. The processing that results from running a program specified in a routing entry. Most jobs have only one routing step.

routing table. In SNADS, a list of entries in a table that the system uses to route a message or electronic mail to a user on the system. Each entry is made up of a destination group name (such as a department or organization) and a destination element name (the user ID of each person in that department or organization).

RU chain. In SNA, a set of related request or response units that are transmitted consecutively on a particular normal or expedited data flow. See also *bracket*.

run time. The time during which the instructions of a computer program are run by a processing unit.

SAA. See *Systems Application Architecture (SAA)*.

save file. A file allocated in auxiliary storage that can be used to store saved data on disk (without requiring diskettes or tapes), to do I/O operations from a high-level language program, or to receive objects sent through the network. The system-recognized identifier for the object type is *FILE.

save storage. An operation that copies (sector by sector) all permanent data from configured disk units to tape.

save system authority. A special authority that allows the user to save and restore all objects on the system and free storage of all objects on the sy

save-while-active operation. An operation that the user runs to save objects while application programs that change the objects are running. Contrast with *dedicated save operation*.

SBCS. See *single-byte character set (SBCS)*.

scheduled job. (1) A batch job that becomes eligible to run at a specified date and time. (2) A batch job that is submitted with a value other than *CURRENT for the schedule date and schedule time parameters.

scope. The extent to which the semantic effects of language statements reach. The scope may be to the job or to the activation group.

scope of control. In OSI, a synonym for *management domain*.

SCS. See *SNA character string (SCS)*.

SDLC. See *synchronous data link control (SDLC)*.

search index. (1) An index of related topics that can be searched or browsed through the InfoSeeker function. The system-recognized identifier for the object type is *SCHIDX. (2) See *InfoSeeker*.

search value. User-defined information that is used to either make a list of filed documents with similar document details or content, or to find a directory entry.

secondary file. (1) For certain types of join operations using Query/400, all files except the first file that are joined in a query definition for the purpose of getting data. (2) In the DDS for a join logical file, any physical file, other than the first physical file, that is specified on the JFILE keyword. Contrast with *primary file*.

secondary language. One or more additional national languages that can be installed on the system to display and print information.

secondary system name. An alternative system name that can be used to identify an AS/400 system in a SNADS network.

secondary system name table. In SNADS, the table containing all the system names that can be used to identify the local system for distributions arriving on the system.

security administrator authority. A special authority that allows a user to add users to the system distribution directory, to create and change user profiles, to add and remove access codes, and to perform office tasks, such as delete documents, folders, and document lists, and change

security officer. A person assigned to control all of the security authorizations provided with the system. A security officer can, for example, remove password or resource security; or add, change, or remove security information about any system user.

select/omit field. A field in a logical file record format whose value is tested by the system to determine if records including that field are to be used. The test is a comparison with a constant, the contents of another field, a range of values, or a list of values; and the record is either selected or omitted as a result of the test.

sequence. To arrange in order.

sequence number. (1) The number of a record that identifies the record within the source member. (2) A field in a journal entry that contains a number assigned by the system. This number is initially 1 and is increased by 1 until the journal is changed or the sequence number is reset by the user.

sequential processing. A method of processing in which records are read, written to, or deleted in the order determined by the value of the key field.

server. A computer that shares its resources with other computers in the network.

service access point (SAP). A logical address that allows a system to route data between a remote device and the appropriate communications support.

service authority. A special authority that allows the user to perform the alter function in the service functions.

service level. One of the four levels of service (fast, status, data high, or data low) that determines if a distribution is put on the normal or priority distribution queue. See also *distribution service level*.

service processor. The logic that contains the processor function to start the system processor and handle error conditions.

service program. A bound program that performs utility functions that can be called by other bound programs. See also *bound program*.

service provider. In SAA SystemView System Manager/400, the AS/400 system used to provide problem-handling support to another AS/400 system or systems connected to it by communications lines. The service provider may also be the alert focal point in a network.

service requester. In SAA SystemView System Manager/400, the AS/400 system with a program or equipment problem that requires and asks for problem-handling support from another AS/400 system in a network.

session. (1) The length of time that starts when a user signs on at a display station and ends when the user signs off. (2) In PC Support/400, the logical connection between the host system and a personal computer or printer. (3) In communications, the logical connection by which a program or device can communicate with a program or device at a remote location. See also *conversation* and *transaction*. (4) In RJE, the activity of all tasks within a single AS/400 system communicating with a single host system. (5) In SNA, a logical connection between two network locations that can be started, tailored to provide various connection protocols, and stopped, as requested. Each session is uniquely identified in a header by a pair of network addresses identifying the origin and destination of any transmission exchanged during the session. See also *half-session*. (6) In 3270 emulation, the activity that occurs on the communications line between the time that the user enters the command to start emulation and the time the user ends the emulation job.

session connection. In OSI, a connection between two nodes that enables them to communicate at the session layer.

SEU. See *source entry utility (SEU)*.

severity code. A number that indicates how important a message is. The higher the number, the more serious the condition.

shared storage pool. A storage pool that can be shared by more than one subsystem.

shared-weight sort sequence. A sort sequence in which some graphic characters in the sequence may have the

same weight as some other characters in the sequence. Those with the same weight will sort together as if they were the same character.

shift. A keyboard action to allow uppercase or other characters to be entered.

short format. In binary floating-point storage formats, the 32-bit representation of a binary floating-point number, not-a-number, or infinity. Contrast with *long format*.

simplex. In AFP support, pertaining to printing on only one side of the paper. Contrast with *duplex*.

Simplified Chinese. The Chinese character set that has been simplified by reducing the number of strokes in common characters and deleting complicated variants. Simplified Chinese characters are used primarily in the People's Republic of China.

single precision. (1) The specification that causes the floating-point value to be stored (internally) in the short format.

single-byte character set (SBCS). A character set in which each character is represented by a one-byte code. Contrast with *double-byte character set*.

SNA. See *Systems Network Architecture (SNA)*.

SNA character string (SCS). In SNA, a data stream composed of EBCDIC controls, optionally intermixed with end-user data, which is carried within a request/response unit.

SNA distribution services (SNADS). An IBM asynchronous distribution service that defines a set of rules to receive, route, and send electronic mail in a network of systems.

SNA upline facility (SNUF). The communications support that allows the AS/400 system to communicate with CICS/VS and IMS/VS application programs on a host system. For example, DHCF communicates with HCF and DSNX communicates with NetView Distribution Manager.

SNADS. See *SNA distribution services (SNADS)*.

SNADS receiver. A user-configured (using the ADDCMNE command) batch job that is started in the subsystem specified on the communications entry when the system receives SNADS distribution from a sending system in the SNADS network.

SNADS router. A system-provided batch job that runs in the QSNADS subsystem and routes distributions to the configured distribution queue. See also *SNADS receiver* and *SNADS sender*.

SNADS sender. A user-configured (by using the CFGDSTSRV command to add the SNADS distribution queue) batch job that is started in the QSNADS subsystem, and sends distributions to another system in the SNADS network. Contrast with *SNADS receiver*.

SNA/File Services (SNA/FS). A service that allows files to be fetched, moved, and stored at nodes in a SNADS network. SNA/FS provides name structure and version identification mechanisms that uniquely identify files in a network.

SNA/FS. See *SNA/File Services (SNA/FS)*.

SNUF. See *SNA upline facility (SNUF)*.

sort sequence. The order in which graphic characters are arranged during sort operations and other comparisons.

sort sequence table. A table containing the order in which characters are arranged within the computer for sorting, combining, or comparing.

source debugger. A tool for debugging Integrated Language Environment (ILE) programs by displaying a representation of their source code. Contrast with *symbolic debugger*.

source entry utility (SEU). A function of the AS/400 Application Development Tools licensed program that is used to create and change source members.

source file. A file of programming code that is not compiled into machine language. A source file can be created by the specification of FILETYPE(*SRC) on the Create command. A source file can contain source statements for such items as high-level language programs and data description specifications. Contrast with *data file*.

source listing. A portion of a compiler listing that contains source statements and, optionally, test results.

source member. A member of a database source file that contains source statements, such as AS/400 BASIC, 36018dh, SAA C/400, SAA COBOL/400, SAA FORTRAN/400, SAA RPG/400, or DDS statements.

source physical file. In PC Support/400, a file that stores text or source statements instead of data.

source program. (1) A set of instructions that are written in a programming language and must be translated to machine language before the program can be run. (2) In communications, the program that starts a session with a remote system. Contrast with *target program*.

source service access point (SSAP). In SNA and TCP/IP, a logical address that allows a system to send data to a remote device from the appropriate communications support. See also *destination service access point (DSAP)*.

source system. (1) In communications, the system that issues a request to establish communications with another system. (2) In DDM, the system on which an application program issues a request to use a remote file.

special authority. The types of authority a user can have to perform system functions, including all object authority, save system authority, job control authority, security administrator authority, spool control authority, and service authority.

special character. (1) In REXX, a token that acts as a delimiter when found outside a literal string. Special characters include the comma (,), semicolon (;), colon (:), right parenthesis ()), left parenthesis ((), and the individual characters from the operators.

specific authority. The types of authority a user can be given to use the system resources, including object authorities and data authorities.

sphere of control. In SNA, a collection of network node control points for which another system is acting as a focal point. This collection includes both control points explicitly defined by the customer, if the controlling system is a primary focal point, and control points assumed by the system if the controlling system is a default focal point.

spool. The system function of putting files or jobs into disk storage for later processing or printing.

spool control authority. A special authority that allows the user to perform spooling functions, such as display, delete, hold, and release spooled files on the output queue for himself and other users. This authority also allows the user to change the spooled file attributes, such as the printer used to print the file. d

spooled file. A file that holds output data waiting to be processed, such as information waiting to be printed. Also known as *spooled output file*.

spooled output file. See *spooled file*.

spooling reader. The general name to refer to the function of the diskette reader and the database reader.

spooling subsystem. A part of the system that provides the operating environment for the programs that read jobs onto job queues to wait for processing and write files from an output queue to an output device. IBM supplies one spooling subsystem: QSPL.

spooling writer. The general name to refer to the function of the diskette writer and printer writer.

SQL. See *Structured Query Language (SQL)*.

SQL descriptor area (SQLDA). A set of variables that are used in the processing of certain SQL statements. The SQLDA is intended for dynamic SQL programs.

SQLDA. See *SQL descriptor area (SQLDA)*.

SRC. See *system reference code (SRC)*.

| **SRM**. See *system resources manager (SRM)*.

SS. See *start-stop (SS)*.

| **stale**. In SAA Application Development Manager/400, pertaining to a part whose source and related parts have changed since the part was last built. Contrast with *current*.

start-of-text (STX) character. In binary synchronous communications, a transmission control character used to begin a logical set of records that will be ended by the end-of-text character or end-of-transmission-block character.

start-stop (SS). Pertaining to asynchronous communications line control that uses start signals and stop signals to control the transfer of data over a communications line. Each group of signals representing a character is preceded by a start signal and followed by a stop signal.

| **static program call**. A connection among programs during binding (program creation time). See also *binding*, *bound program*, and *service program*.

static storage. In OS/400 application programming interfaces, an area that is created when a program is activated. Each routine can have either automatic storage or static storage. Within static storage, variables are defined. Contrast with *automatic storage*.

station address. A 2-character hexadecimal value from 01 to FE. For a primary controller, it is called the SDLC station address; for a secondary controller, it is called the remote system address.

step. To cause a computer to run one operation.

storage pool. A logical division of storage reserved for processing a job or group of jobs.

store. To put or keep data in a storage device.

string. (1) A group of auxiliary storage devices connected in a series on the system. The order and location in which each device is connected to the system determines the physical address of the device. (2) In REXX, a sequence of elements of the same nature, such as characters considered as a whole; for example, character string, binary string, and hexadecimal string.

| **subfolder**. A folder that is in another folder. For example, if folder A contains folder B and folder B contains folder C, then B and C are subfolders of A because the folder path for each begins with A (A/B/C). See also *folder path* and *root folder*.

subprogram. A called program. A subprogram is combined with the calling program at run time to produce a run unit and is below the calling program in the program stack.

subroutine. (1) A group of instructions within another group of instructions that can be called by a program or another subroutine. (2) In data communications, a group of statements in a program that can be run several times in that program. (3) In REXX, an internal, built-in, or external routine called by the CALL instruction that may or may not return a result string. If a subroutine returns a result string, a subroutine can also be called by a function call, in which case it is being called as a function.

subscript. A symbol, number, or letter written immediately below and to the right or left of another character. For example, the number 2 in the chemical formula for water, H_2O, is a subscript.

substitution variable. A variable used to pass information, such as a file name, for use in a message.

substring. A part of a character string.

subsystem. An operating environment, defined by a subsystem description, where the system coordinates processing and resources.

subsystem description. A system object that contains information defining the characteristics of an operating environment controlled by the system. The system-recognized identifier for the object type is *SBSD.

superscript. A symbol, number, or letter written immediately above and to the right or left of another character. For example, a footnote can be identified in text with a superscript number.

switched line. In data communications, a connection between computers or devices that is established by dialing.

switched network backup (SNBU). A modem feature that allows a nonswitched line to be used alternatively as a switched line or allows a switched line to be used as a nonswitched line depending on the characteristics of the modem.

switched virtual circuit (SVC). (1) A virtual circuit that is requested by a virtual call. It is released when the virtual circuit is cleared.

symbol. In REXX, any combination of alphabetic or numeric characters (A-Z, a-z, or 0-9) and the characters @, #, $, ¢, ., !, ?, and _.

symbol set. In AS/400 Business Graphics Utility, a supplied character set used for text strings on charts; for example, headings, legend text, labels, and notes.

symbolic debugger. A tool that aids in the debugging of programs written in certain high-level languages. Contrast with *source debugger*.

synchronization (SYN) character. In binary synchronous communications, the transmission control character that provides a signal to the receiving station for timing the characters received.

synchronous data link control (SDLC). (1) A form of communications line control that uses commands to control the transfer of data over a communications line. (2) A communications discipline conforming to subsets of the Advanced Data Communication Control Procedures (ADCCP) of the American National Standards Institute (ANSI) and High-Level Data Link Control (HDLC) of the International Organization for Standardization (ISO), for transferring synchronous, code-transparent, serial-by-bit information over a communications line. Transmission exchanges may be duplex or half-duplex over switched or nonswitched lines. The configuration of the connection may be point-to-point, multipoint, or loop. Compare with *binary synchronous communications (BSC)*.

synchronous processing. A series of operations that are done as part of the job in which they were requested; for example, calling a program in an interactive job at a work station. Contrast with *asynchronous processing*.

synonym. One of two or more words of the same language that have the same or nearly the same meaning.

syntax checking. A function of the system, a compiler, the BASIC interpreter, or SEU that checks individual statements for errors in the structure of the statement.

system ASP. The auxiliary storage pool where system programs and data reside. It is the storage pool used if a storage pool is not defined by the user. See also *auxiliary storage pool* and *user ASP*.

system ASP. The auxiliary storage pool where system programs and data reside. The system ASP (ASP1) always exists. See also *auxiliary storage pool (ASP)* and *user ASP*.

system date. The date assigned in the system values when the system is started.

system distribution directory. A list of user IDs and identifying information, such as network addresses, used to send distributions.

system library. The library shipped with the system that contains objects, such as authorization lists and device descriptions created by a user; and the licensed programs, system commands, and any other system objects shipped with the system. The system identifier is QSYS.

system name. (1) An IBM-supplied name that uniquely identifies the system. It is used as a network value for certain communications applications such as APPC. (2) An IBM-defined name that has a predefined meaning to the COBOL compiler. System names include computer names, language names, device names, and function names.

system object. A machine object classification. Any of the machine objects shipped with the system or any of the operating system objects created by the system.

system pointer. A pointer that contains addressability to a machine interface system object.

system processor. The logic that contains the processor function to translate and process the OS/400 control language commands and programming language statements.

system reference code (SRC). The characters that identify the name of the unit that detected the condition and the reference code that describes the condition.

system resources manager (SRM). A group of programs that controls the use of system resources, such as programs, devices, and storage areas that are assigned for use in jobs.

system security. A system function that restricts the use of files, libraries, folders, and devices to certain users.

system service tools (SST). The part of the service function used to service the system while the operating system is running.

system services control point (SSCP). A focal point within an SNA network for managing the other systems and devices, coordinating network operator requests and problem analysis requests, and providing directory routing and other session services for network users.

system unit. A part of a computer that contains the processing unit, and may contain devices such as disk units and tape units.

system value. Control information for the operation of certain parts of the system. A user can change the system value to define his working environment. System date and library list are examples of system values.

System/36 environment. A function of the operating system that processes most of the System/36 operator control language (OCL) statements and procedure statements to run System/36 application programs and allows the user to process the control language (CL) commands.

System/36 object. A configuration description in System/36 terms that defines the System/36 environment. The system-recognized identifier for the object type is *S36.

System/38 environment. A function of the operating system that processes most of the System/38 control language (CL) statements and programs to run System/38 application programs.

Systems Application Architecture (SAA). Pertaining to an architecture defining a set of rules for designing a common user interface, programming interface, application programs, and communications support for strategic operating systems such as the OS/2, OS/400, VM, and MVS operating systems.

systems management. In SAA SystemView System Manager/400, all of the actions and procedures that accomplish the business support activity of making information systems services available. Information systems services include host, application, network, and data services.

systems management application. In OSI, an application that provides systems management services. See also *application* and *systems management*.

Systems Network Architecture (SNA). In IBM networks, the description of the layered logical structure, formats, protocols, and operational sequences that are used for transmitting information units through networks, as well as controlling the configuration and operation of networks.

Systems Network Architecture distribution services. See *SNA distribution services (SNADS)*.

table. (1) An orderly arrangement of data in rows and columns that can contain numbers, text, or a combination of both. The system-recognized identifier for the object type is *TBL. See also *translation table*. (2) In CSP/AE, a collection of related data items arranged as a two-dimensional array of columns and rows that can be used in verifying map inputs or identifying related factors for standard calculations. The system-recognized identifier is *CSPTBL.

tag. In UIM, the actual statements of the UIM tag language. Tags describe the actions, format, and data of the panel. Tags are used to define the formatting of help information.

tangent. In GDDM, the single point at which a straight line meets a curve or surface.

tape drive. A device used to move the tape and read and write information on magnetic tapes.

tape file. A device file to support a tape device.

tape mark. A unique mark written on the tape to distinguish file boundaries.

tape reel. A round device on which magnetic tape is wound.

tape unit. The physical enclosure containing the tape drive.

tape volume. A single reel of magnetic tape.

target. (1) In advanced program-to-program communications, the program or system to which a request for processing is sent. (2) In DDM, the remote system where the request for a file is sent. (3) In SEU, a line command, such as B (Before) or A (After), that specifies the destination for other line commands such as C (Copy) or M (Move).

target program. (1) In communications, the program that is started on the remote system at the request of the source system. (2) In display station pass-through, a program that runs on the remote system.

target system. (1) The system that receives a request from another system to establish communications. (2) In a distributed data management (DDM) network, the system that receives a request from an application program on another system to use one or more files located on the target system.

TCP/IP. See *Transmission Control Protocol/Internet Protocol (TCP/IP)*.

technical information exchange (TIE). A part of the electronic customer support function that allows a user to send files to and receive files from an IBM support system, and to search for information on an IBM support system. The files are sent and received through an IBM Information Network.

temporary library. A library that is automatically created for each job to contain temporary objects that are created by the system for that job. The objects in the temporary library are deleted when the job ends. The system name for temporary library is QTEMP.

temporary objects. Objects, such as data paths or compiler work areas, that are automatically deleted by the system when the operating system is loaded.

terminal equipment (TE). In an ISDN, data terminal equipment (DTE) that provides the function necessary for the operation of the access protocols by the user

terminal equipment 1 (TE1). Data terminal equipment (DTE) with integrated ISDN support. In an ISDN, the AS/400 system is a TE1.

terminal equipment 2 (TE2). Data terminal equipment (DTE) without an ISDN interface. To communicate with other equipment through an ISDN, this equipment must have the protocol converted to one that can be recognized by the network. For example, a 7820 ISDN terminal adapter may be used.

test library. A user-defined library used for debugging operations that does not contain objects needed for normal processing. Contrast with *production library*.

text transparency. In BSC, a method of sending and receiving data containing any or all of the 256 character combinations in EBCDIC in specific bit patterns, including transmission control characters. Transmission control characters sent in the data are treated as specific bit patterns, unless they are preceded by the DLE control character.

threshold. A level set in the system at which a message is sent or an error-handling program is called. For example, in a user auxiliary storage pool, the user can set the threshold level in the system values, and the system notifies the system operator when that level is reached.

TIE. See *technical information exchange (TIE)*.

time slice. The amount of processor time (specified in milliseconds) allowed for a job before other waiting jobs of equal priority are allowed to process data.

token. (1) A predefined message or character pattern that gives the receiver of the token the permission to transmit information. (2) The unit of low-level syntax from which REXX clauses are built. Tokens include literal strings, operator characters, and special characters.

token-ring network. A local area network that sends data in one direction throughout a specified number of locations by using the symbol of authority for control of the transmission line, called a token, to allow any sending station in the network (ring) to send data when the token arrives at that location.

topology. The schematic arrangement of the links and nodes of a network.

topology database. The representation of the current topology of the intermediate routing portion of the APPN network. The network topology database contains entries for network nodes and the transmission groups interconnecting them. Each entry describes the current characteristics of the node or transmission group that it represents. The topology database is used to determine the preferred session route between two end nodes for a given class of service.

total record. In RPG, an output record written after a group of detail records. Total records generally contain data that is the result of calculations performed on the information in a group of detail records.

trace. In REXX, a means of tracking the interpretation of a program. Tracing is primarily used for debugging.

Traditional Chinese. The Chinese character set expressed in traditional form. Traditional Chinese characters are used in Taiwan, Hong Kong and some other parts of the world.

transaction. (1) An item of business, for example, the handling of customer orders and customer billing. (2) In communications, an exchange between a program on a local system and a program on a remote system that accomplishes a particular action or result. See also *conversation* and *session*. (3) In performance, a unit of work used to express the throughput of a workload or to request the estimated response time. An interactive transaction is the work done by the system when the Enter key or a function key is pressed. A noninteractive transaction is defined in terms of resource activity used by the noninteractive jobs.

translation table. An object that contains a set of hexadecimal characters used to translate one or more characters of data. The table can be used for translation of data being moved between the system and a device. For example, data stored in one national language character set may need to be displayed or entered on display devices that support a different national language character set. The table can also be used to specify an alternative collating sequence or field translation functions. The system-recognized identifier for the object type is *TBL.

transparent text mode. In binary synchronous communications, a method of transmission in which only transmission control characters preceded by the DLE control character are processed as transmission control characters.

TRLAN. Abbreviation in the commands, parameters, and options for IBM Token-Ring Network. See also *token-ring network*.

twinaxial data link control (TDLC). A communications function that allows personal computers, which are attached to the work station controller by twinaxial cable, to use advanced program-to-program communications (APPC) or advanced peer-to-peer networking (APPN).

UIM. See *user interface manager (UIM)*.

unformatted. Pertaining to something that is not defined, organized, or arranged in a required manner.

unique-weight sort sequence. A sort sequence in which each graphic character in the sequence has a weight different from the weight of every other graphic character in the sequence.

unit. The defined space within disk units that is addressed by the system.

unlink. In IDDU, to remove the association between a database file on disk and a file definition in a data dictionary.

unload. To rewind tape past the beginning-of-tape marker.

update authority. A data authority that allows the user to change the data in an object, such as a journal, a message queue, or a data area.

upline. Pertaining to controllers that are above devices, and lines that are above controllers in a communications configuration.

use authority. An object authority that allows the user to run a program or to display the contents of a file. Use authority combines object operational authority and read authority.

user ASP. One or more storage units used to isolate some objects from the other objects stored in the system ASP. User ASPs are defined by the user. See also *auxiliary storage pool (ASP)* and *system ASP*.

user class. The classification of a user by the system task, such as security officer, security administrator, programmer, system operator, and user. Each user class has a set of special authorities depending on the security level of the system. The user class determines which options are shown on the IBM-supplied menus.

user default. In SAA OfficeVision/400, the user-defined values used for mail, calendars, and word processing if no value is specified.

user ID. See *user identification (user ID)*.

user identification (user ID). (1) The name used to associate the user profile with a user when a user signs on the system. (2) The first part of a two-part network name used in the system distribution directory and in the office applications to uniquely identify a user. The network name is usually the same as the user profile name, but does not need to be. See also *common user identification (common user ID)*.

user index. In OS/400 application programming interfaces, an object that provides a specific order for byte data according to the value of the data. User index objects reside in the user domain. The system-recognized identifier for the object type is *USRIDX.

user interface manager (UIM). A function of the operating system that provides a consistent user interface by providing comprehensive support for defining and running panels (displays), dialogs, and online help information.

user message queue. A user-created object used to receive messages sent from the system, other users, and application programs.

user profile. An object with a unique name that contains the user's password, the list of special authorities assigned to a user, and the objects the user owns. The system-recognized identifier for the object type is *USRPRF.

user profile name. The name or code that the system associates with a user when he or she signs on the system. Also known as user ID.

user queue. In OS/400 application programming interfaces, an object consisting of a list of messages that communicate information to other application programs. The system-recognized identifier for the object type is *USRQ.

user space. In OS/400 application programming interfaces, an object consisting of a collection of bytes that can be used for storing any user-defined information. The system-recognized identifier for the object type is *USRSPC.

user table. A list of user IDs authorized to an AS/400 finance job.

validity checking. To verify the contents of a field.

value. (1) Data (numbers or character strings) entered in any entry field, and data supplied in parameters of CL commands. (2) The smallest unit of data manipulated by the Structured Query Language. (3) In query management, a quantity assigned to a keyword or variable associated with a query command. If the keyword is part of the command string, its value is separated from it with an equal sign (=). If the keyword is an argument on the extended interface, its value will also be an argument.

vary off. To make a device, controller, line, or network interface unavailable for its normal, intended use.

vary on. To make a device, controller, line, or network interface available for its normal, intended use.

vector symbol set (VSS). In GDDM, a set of characters each of which is treated as a small picture and is described by a sequence of lines and arcs. Characters in a vector symbol set can be drawn to scale, rotated, and positioned precisely.

version. (1) A separate IBM licensed program, based on an existing IBM licensed program, that usually has significant new code or new function. Each version has its own license, terms, conditions, product type number, monthly charge, doc-

umentation, test allowance (if applicable), and programming support category. The numbering of versions starts with Version 2. See also *modification level* and *release*. (2) In the SAA Application Development Manager/400 product, a separate program or release, new or based on an existing application, that contains significant new code or function.

vertical licensed internal code (VLIC). Programming that defines logical operations on data. The vertical licensed internal code translates the machine interface (MI) instructions.

view. (1) The form in which an object is presented. A choice in the action bar that a user selects to look at an object from various perspectives is an example of a view. (2) A choice in the action bar that a user selects to look at an object from various perspectives.

virtual device. A device description that does not have hardware associated with it. It is used to form a connection between a user and a physical work station attached to a remote system. A virtual device can be a virtual display station or a virtual printer.

virtual storage (VS). An addressing scheme that allows external disk storage to appear as main storage.

virtual work station controller. A work station controller description that has the characteristics of a locally attached work station controller but does not exist as hardware.

VLIC. See *vertical licensed internal code (VLIC)*.

VM. See *virtual machine (VM)*.

volume. A storage medium that is put on or taken off the system as a unit, for example, magnetic tape or diskette.

volume label. The first 80 bytes on a standard tape used to identify the tape volume and its owner. This area contains VOL1 in the first four positions.

volume statistics. Statistical information about the activity on a tape, diskette, or cartridge volume including statistics about the session (such as the number of read, write, and retry operations), and "lifetime" (accumulated) statistics (such as the number of read and write errors and the number of bytes read and written).

volume table of contents (VTOC). An area on a disk or diskette that describes the location, size, and other characteristics of each file, library, and folder on the disk or diskette.

whole number. In REXX, an integer or a number that has a zero decimal part. Whole numbers are not usually expressed by the language processor in exponential notation.

wide area network (WAN). A data communications network designed to serve an area of hundreds or thousands of miles—for example, public and private packet-switching networks, and national telephone networks.

word. In REXX, a sequence of characters that do not include any blanks. Words may be used as units for manipulation during parsing and by many built-in functions.

work on behalf of. Pertaining to the function that allows users to temporarily access documents, folders, or mail that another user is authorized to except those items that are marked personal. Tasks performed by a user working on another user's behalf produce the same results as if the original user performed the task. For example, if user A creates a new object while working on behalf of user B, user B is the owner of the object.

work station. A device used to transmit information to or receive information from a computer, for example, a display station or printer.

work station controller (WSC). An I/O controller card in the card enclosure that provides the direct connection of local work stations to the system.

work station entry. An entry in a subsystem description that specifies the work stations from which users can sign on to the subsystem or from which interactive jobs can transfer to the subsystem.

write operation. An output operation that sends a processed record to an output device or output file.

writer. (1) The part of the operating system spooling support that writes spooled files to an output device independently of the program that produced the output.

writing. The action of making a recording of data on an external storage device or other data medium.

X.25. A CCITT Recommendation that defines the physical level (physical layer), link level (data link layer), and packet level (network layer) of the OSI reference model. An X.25 network is an interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE) operating in the packet mode, and connected to public data networks by dedicated circuits. X.25 networks use the connection-mode network service.

zoned decimal format. A format for representing numbers in which the digit is contained in bits 4 through 7 and the sign is contained in bits 0 through 3 of the least significant byte; bits 0 through 3 of all other bytes contain 1's (hex F). For example, in zoned decimal format, the decimal value of +123 is represented as 1111 0001 1111 0010 1111 0011. Synonymous with *unpacked decimal format*.

12-hour clock. A clock that keeps time from 12:00 a.m. (midnight) to 12:00 p.m. (noon), and from 12:00 p.m. (noon) to 12:00 a.m. (midnight).

24-hour clock. A clock that keeps time from 0000 (midnight) to 1200 (noon), and from 1200 (noon) to 2400 (midnight).

3270 device emulation. The operating system support that allows an AS/400 system to appear as a 3274 Control Unit in a BSC multipoint network or an SNA network.

3270 display emulation. The function of the operating system 3270 device emulation support that converts 3270 data streams intended for a 3278 display station into data streams that can be recognized by a display station attached to the AS/400 system.

3270 printer emulation. The function of the operating system 3270 device emulation support that converts 3270, DSC, and SCS data streams intended for a 328X printer into data streams that can be recognized by a printer attached to the AS/400 system.

400/REXX interpreter. The language processor of the OS/400 licensed program that processes procedures and programs written in the REXX language.

5250 emulation. Any one of many licensed programs that allow a personal computer to perform like a 5250 display station or printer, and use the functions of an AS/400 system.

5494 Remote Control Unit. A control unit that can attach up to 56 work stations and allows either twinaxial or token-ring attachment.

Index

Special Characters

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

Customer Satisfaction Feedback

Application System/400
Programming:
Control Language Reference
Version 2

Publication No. SC41-0030-02

Overall, how would you rate this manual?

	Very Satisfied	Satisfied	Dissatis-fied	Very Dissatis-fied
Overall satisfaction				

How satisfied are you that the information in this manual is:

Accurate				
Complete				
Easy to find				
Easy to understand				
Well organized				
Applicable to your tasks				
THANK YOU!				

Please tell us how we can improve this manual:

May we contact you to discuss your responses? __ Yes __ No

Phone: (____) _____ Fax: (____) _____

To return this form:

- Mail it
- Fax it
  United States and Canada: **800+937-3430**
  Other countries: **(+1)+507+253-5192**
- Hand it to your IBM representative.

Note that IBM may use or distribute the responses to this form without obligation.

Name

Address

Company or Organization

Phone No.

Customer Satisfaction Feedback
SC41-0030-02

IBM®

Cut
Alon

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN DEPT 245
IBM CORPORATION
3605 HWY 52 N
ROCHESTER MN 55901-9986

Fold and Tape | **Please do not staple** | Fold and Tape

SC41-0030-02

Cut
Alo

Customer Satisfaction Feedback

Application System/400
Programming:
Control Language Reference
Version 2

Publication No. SC41-0030-02

Overall, how would you rate this manual?

	Very Satisfied	Satisfied	Dissatis-fied	Very Dissatis-fied
Overall satisfaction				

How satisfied are you that the information in this manual is:

Accurate				
Complete				
Easy to find				
Easy to understand				
Well organized				
Applicable to your tasks				

THANK YOU!

Please tell us how we can improve this manual:

May we contact you to discuss your responses? __ Yes __ No

Phone: (____) _____ Fax: (____) _____

To return this form:

- Mail it
- Fax it
  United States and Canada: **800+937-3430**
  Other countries: **(+1)+507+253-5192**
- Hand it to your IBM representative.

Note that IBM may use or distribute the responses to this form without obligation.

Name

Address

Company or Organization

Phone No.

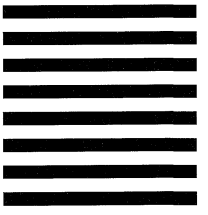
Customer Satisfaction Feedback
SC41-0030-02

IBM®

Cut o
Along

Fold and Tape | **Please do not staple** | Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN DEPT 245
IBM CORPORATION
3605 HWY 52 N
ROCHESTER MN 55901-9986

Fold and Tape | **Please do not staple** | Fold and Tape

SC41-0030-02

Cut
Alon

Program Number: 5738-SS1

Printed in Denmark by Bonde's

SC41-0030-02