How to contact Borland

Borland offers a variety of services to answer your questions about InterBase:

Customer support

Registered customers can call the InterBase customer support line at 1-800-437-7367 from within the United States or Canada with their questions about InterBase. International customers can call 408-431-5400.

General classes are also available for InterBase training and consultants are available for individual customer site needs.

Marketing information

If you are not already an InterBase customer and want information about this product, call the product information line at 1-800-245-7376.

Documentation feedback

We welcome your feedback about errors or missing topics in the InterBase documentation.

Table of Contents

Overview

This document describes changes that have occurred since InterBase Version 3.2:

- V3.3 features and changes, including Paradox 4.0 SQL Link access to InterBase databases and international support for InterBase
- Software restrictions and suggested workarounds, when available
- Bug fixes
- Documentation corrections

All release notes and addenda previous to InterBase Version 3.3 are now included in *InterBase 3.0 to 3.2 Release Notes*.

The following two documents: *InterBase Version 3.2 Documentation Corrections* and *InterBase Previous Versions Documentation Corrections*, have been discontinued. Most of the documentation corrections and clarifications that were in these documents have been included in the June 1992 reprint of the InterBase V3.0 documentation set. This reprint does not contain any features that were added following the original V3.0 release. All documentation corrections and clarifications since V3.2 are in this document.

V3.3 Features

New Language Support

COBOL

InterBase now supports Microfocus COBOL on the Sun4, Data General, IBM (AIX), and HP 700/800 platforms. In addition, VAX/VMS COBOL is supported on that platform. For information on using COBOL on these platforms, refer to the appropriate InterBase installation guide.

FORTRAN

Now supported on the IBM (AIX) platform. Refer to *InterBase 3.3 - Installing and Running on UNIX*, for information on using FORTRAN on this platform.

Platform and OS Support

For information on V3.3 platforms, operating systems, and language support, refer to the chart at the end of this release note. Note that the following support has been discontinued:

	Discontinued Support
Platform	VAX, VAXstation/server running VAX/Ultrix, Sun-3
OS	HP-UX 7.*n*, AIX 3.1, SCO 3.2.2, SunOS 4.1.1
Language	BASIC and PL/1 on VAX/VMS

InterBase Support for Paradox 4.0 SQL Link

InterBase V3.3 allows remote PC clients running Paradox 4.0 SQL Link to access InterBase V3.3 databases on UNIX servers. To connect to InterBase, PC users should consult the Paradox 4.0 documentation on this subject. Because PC systems do not traditionally provide login and password security features for user authentication, InterBase V3.3 implements a database-level authentication scheme, described below.

User Authentication Requirements

When a remote PC client attempts to attach a database, the client is required to enter a valid user name and password before the attach can succeed. The password is encrypted for transmission over the network, and then both the user name and password are verified against records stored in */usr/interbase/isc.gdb*, the authentication database automatically installed with InterBase V3.3.

Note

On Apollo systems the authentication database location is */interbase/isc.gdb*. On VAX/VMS systems, the database is stored in the location pointed to by the SYS$MANAGER logical.

As shipped, *isc.gdb* does not contain site-specific login and password information. To enable remote PC clients to access your InterBase databases, you must add user identification records to *isc.gdb*. At a minimum, each record should contain two fields:

- a user name
- a password

The **gsec** security utility, installed with InterBase V3.3, allows you to add, modify, delete, and view user identification records in *isc.gdb*.

Using **gsec** to Maintain the Password Database

The **gsec** utility lets you change the general InterBase user information in *isc.gdb*. You can use **gsec** to add, modify, delete, and view user identification records.

To run **gsec** you must be logged in as superuser.

gsec runs in two modes: interactive, and direct. To run **gsec** interactively, type `gsec` at the system prompt. In interactive mode, **gsec** prompts for the command to execute, as well as any necessary parameters. For example, to add a new user to the database, the **add** command requires the name of the user to add, while the **modify** command requires both a user name and at least one additional parameter which specifies what part of the user record to modify.

To run a **gsec** command directly, enter the entire command, including a user name and additional parameters as required. The complete syntax for **gsec** is described below.

gsec Syntax

Generic **gsec** syntax for all **gsec** tasks is:

```
gsec [<command>] [<username>] [<param> [<param>...]
```

where:

- *<command>* is one of:

Table 1. gsec Commands

Command	Purpose
-add	Add a new user record
-delete	Delete a user record
-modify	Change a user record
-display	Display one or more user records
-help	Display command explanations
quit	Interactive mode only--exit **gsec**

Notes

1. Commands need only be long enough to distinguish them from other **gsec** commands and parameters. For example, **-display** may be shortened to **-di**.

2. In interactive mode, do *not* precede commands with the switch character (-).

- *<username>* is the actual name the user should enter to login to a database.
- *<param>* is one or more of:

Table 2. gsec Parameters

Parameter	Purpose	Type
-pw *<password>*	The user's password	string
-uid *<uid>*	A user ID number	long integer
-gid *<gid>*	The user's group ID number	long integer
-proj *<projectname>*	On Apollo, a project name	string
-org *<organizationname>*	On Apollo, an organization name	string
-fname *<firstname>*	The user's actual first name	string
-mname *<middlename>*	The user's middle name	string
-lname *<lastname>*	The user's last name	string
-z	Display **gsec** version number	—

Notes

1. Parameters need only be long enough to distinguish them from other **gsec** commands and parameters. For example, **-proj** may be shortened to **-pr**.

2. *isc.gdb* provides a level of security apart from normal UNIX login and password security features. Therefore, **gsec** parameters which mimic UNIX or platform security features, e.g., **-pw** or **-uid**, need not correspond to existing UNIX information.

Adding a New User to *isc.gdb*

To allow a remote PC client to access InterBase databases from Paradox 4.0 SQL Link, use **gsec** to add the client's user name and password to *interbase*. Before storing a password in *isc.gdb*, **gsec** encrypts it.

To add a user interactively:

1. Log in as superuser.
2. Start **gsec**:

```
% gsec
```

3. At the **gsec** prompt, enter the **add** command. For example, the following command adds user DWISE to *isc.gdb*.

```
GSEC> add dwise -pw boss -uid 1 -fname Daniel -lname Wise
```

To add a user from the command line:

1. Log in as superuser.
2. Enter the desired **add** command. For example, the following command adds user DWISE to *isc.gdb*.

```
% gsec -add dwise -pw boss -uid 1 -fname Daniel -lname Wise
```

Note

For a complete list of **gsec** parameters, see *gsec Syntax*.

Deleting a User from *isc.gdb*

When a remote PC client no longer requires access to InterBase databases, use **gsec** to remove the client's record from *isc.gdb*. The **gsec** syntax required is:

```
% gsec -delete <username>
```

Displaying User Information in *isc.gdb*

To see information for every user in the *isc.gdb* database, type this command:

```
% gsec -display
```

To see information for a single user, the **gsec** syntax is:

```
% gsec -display [<username>]
```

Note

To provide a measure of security, **gsec** does *not* display user passwords.

Modifying User Information in *isc.gdb*

To change or add information about a user already entered in *isc.gdb*, use this **gsec** syntax:

```
% gsec -modify <username> <param> [<param>...]
```

When modifying a user record, you must supply at least one parameter on the command line. Each parameter must be immediately followed by the information to change or add. For example, the **-fname** parameter requires a user's first name:

```
% gsec -modify spike -fname Leslie
```

Creating and Altering Blob Columns in DSQL

In InterBase V3.3 the DSQL **create table** and **alter table** commands support tables with blob columns.

Syntax

The syntax for the **create table** command is:

```
create table <tname> (<colname> blob [ ( [<seglen>]
[,<subtype>] ) ][, ...] )
```

The syntax for adding columns to a table with the **alter table** command is:

```
alter table <tname> add <colname> blob [ ( [<seglen>]
[,<subtype>] ) ] [, ...]
```

The syntax for dropping a column from a table with the **alter table** command is:

```
alter table <tname> drop <colname> [, ...]
```

where:

tname is the name of the table to create.

colname is the name of the column to create in the table.

seglen is an optional parameter indicating the size, in bytes, of the blob segment. InterBase allocates memory for a blob data transfer buffer based on the size of the blob segment. If a command omits *seglen*, InterBase uses a default of 80 bytes.

subtype is an optional parameter specifying the kind of blob column. InterBase uses the blob subtype to determine how to handle blob processing. *subtype* is a short integer ranging from -32768 to 32767. InterBase predefines subtypes 0 through 12 for its own use, although your programs can use InterBase subtypes 0 and 1. To avoid conflict with future expansion of InterBase subtypes, user-defined subtypes should be defined as negative integers, ranging from -32768 to -1. If a command omits *subtype*, InterBase uses a default of 0.

Examples

The following command creates a table named TRAVEL with a blob column called PURPOSE. Because *seglen* is unspecified, segment length defaults to 80 bytes, and because *subtype* is unspecified, the blob subtype is set to 0.

```
create table travel (purpose blob)
```

The next command adds a blob column named ITINERARY to the TRAVEL table, and sets its segment length to 128 bytes. Because *subtype* is unspecified, the blob subtype is set to 0.

```
alter table travel add itinerary blob (128)
```

The following creates a table named VACATION with blob columns named DESCRIPTION and DESTINATION. For the DESCRIPTION column, the blob subtype is set to 1, but because *seglen* is unspecified, segment length defaults to 80 bytes. For the DESTINATION column, segment length is set to 40, and blob subtype is set to 1.

```
create table vacation (description blob (,1), destination
blob (40,1))
```

The next command removes the blob column named DESTINATION from the VACATION table.

```
alter table vacation drop destination
```

Additional Blob and DSQL Information

For additional information about the blob data type, including segment length and blob subtype, see the *Data Definition Guide*. For information about programming with DSQL, see the *DSQL Programmer's Guide*. For other **create table** and **alter table** command options, see the *Programmer's Reference*.

The Extended SQL Descriptor Area (XSQLDA)

InterBase supports a new, extended SQL Descriptor Area (XSQLDA) for use when you write dynamic SQL (DSQL) programs. While programs using the old SQL Descriptor Area (SQLDA) continue to be supported, new DSQL programs should implement the XSQLDA in its place. Eventually, older programs should also be updated to use the XSQLDA.

Programming for the XSQLDA

Programming for the XSQLDA will be little different from programming for the SQLDA. In addition to providing XSQLDA version information, the XSQLDA includes an extended SQLVAR structure (XSQLVAR) for passing parameters to DSQL and receiving **select** output from DSQL. The XSQLVAR allows more complete parameter passing to DSQL, and returns more output information to the host program. Specifically, the XSQLDA allows passage of scale factor, blob subtype, relation name, relation owner name, and relation alias.

Using **gpre** to Preprocess Programs Implementing XSQLDA

To allow for the future addition of alternate SQLDA structures such as the XSQLDA, a new switch, **-sqlda**, has been added to **gpre**. To preprocess programs which use the new XSQLDA, you must use the **-sqlda** switch. The correct syntax is:

```
% gpre -sqlda new <progname>
```

Note

To support the large number of customer programs that already implement the old SQLDA structure, the default behavior of **gpre** is to generate SQLDA calls during preprocessing.

Structure of the XSQLDA

For programming purposes, the structure of the XSQLDA is similar to the structure of the SQLDA. At any single time an XSQLDA contains information about either:

- Input parameters from the host program to DSQL
- The output of **select** requests from DSQL to the host program

An XSQLDA cannot simultaneously contain both input and output information. The following table describes the fields of an XSQLDA:

Table 3. XSQLDA fields

Field Name	Size	Purpose
VERSION	short int	The XSQLDA version number. Your program must set this number to 1.
SQLAID	char [8]	An internal string identifier for the extended descriptor area. Your program must set this to the following 8-byte string: "XSQLDA ".
SQLABC	long int	XSQLDA length indicator.
SQLN	short int	Indicates the maximum number of input parameters or **select** list items in the XSQLDA. This value specifies how many XSQLVAR occurrences that the XSQLDA can hold. Your program sets this number. It must be equal to or greater than the value in SQLD.
SQLD	short int	Indicates either the number of input parameters provided by your program, or the number of fields returned by a DSQL **select** statement. Your program can use this field to: • specify number of parameters to SQL • determine storage allocation for select list items • determine if a prepared statement is a **select** statement Returned **select** information is filled in automatically when you use a **prepare** or **describe** statement.
SQLVAR	sizeof(XSQLVAR)	Describes either: • Each parameter associated with a DSQL statement • Each list item in a DSQL **select** statement There is one SQLVAR for each parameter or select list item.

The following table describes the subfields of each SQLVAR field in an XSQLDA. Most subfields are filled in automatically when you use a **describe** or **prepare** statement.

Table 4. SQLVAR Subfields

Field Name	Size	Purpose
SQLTYPE	short int	Indicates the data type of the input parameter or select list item.
SQLSCALE	short int	Indicates the precision to use for the input parameter.
SQLSUBTYPE	short int	Indicates the data type subtype of the input parameter or select list item. Currently, only blob subtypes may be specified in this field.
SQLLEN	short int	Indicates the byte length of the input parameter or select list item.
SQLDATA	char *	Points to the address of the storage area allocated for the input parameter or select list item.
SQLIND	short *	Points to the address of the indicator variable. This variable permits you to check for null or missing values.
SQLNAME_LENGTH	short int	Indicates the byte length of SQLNAME.
SQLNAME	char [32]	Names the field described by XSQLVAR. This is the name of the select list item.
RELNAME_LENGTH	short int	Indicates the byte length of RELNAME.
RELNAME	char [32]	Names the relation of which the field described by the XSQLDA is a member.
OWNNAME_LENGTH	short int	Indicates the byte length of OWNNAME.
OWNNAME	char [32]	Names the owner of the relation specified in RELNAME.
ALIASNAME_LENGTH	short int	Indicates the byte length of ALIASNAME.
ALIASNAME	char [32]	Provides the alias name for the field specified in SQLNAME.

Note

If you are filling in the XSQLDA with the SQLVAR fields manually, rather than using **describe** or **prepare**, note the addition of the new fields, SQLSCALE and SQLSUBTYPE. These fields will also need to be added.

For a complete discussion of DSQL programming considerations, see the *DSQL Programmer's Guide.*

XSQLDA_LENGTH Macro for C and C++ Programmers

The **include sqlca** statement defines XSQLDA structures and macros. These declarations include a macro, XSQLDA_LENGTH, which determines the amount of memory in bytes required to hold the XSQLDA:

```
#define XSQLDA_LENGTH(n) (sizeof(XSQLDA) + (n-1) *
sizeof(XSQLVAR))
```

For examples of how this macro might be used, see the comparable examples of SQLDA_LENGTH used in the DSQL C program examples in Chapter 2 of the *DSQL Programmer's Guide.*

Remapping of SQLCODE Warnings and Errors

InterBase V3.3 returns a greater range of warning and error values for the SQLCODE. Some mappings of major InterBase error codes to SQLCODE numbers have changed. See *Remapped Major Codes,* below, for a list of SQLCODE remapping.

In addition, many SQLCODE numbers are now mapped to minor InterBase error codes. See *Mapping of Minor Codes,* below, for a list of those SQLCODE mappings.

SQL Programming Considerations

If you intend to move your SQL programs between InterBase and other database management systems, you should limit your error handling routines to using the **whenever** statement and to checking the value of SQLCODE. The **whenever** statement lets you perform actions depending upon the value of the SQLCODE.

You may want to rewrite existing SQL program error handling sections to take advantage of the extended SQLCODE support offered by InterBase V3.3.

Remapped Major Codes

The following table lists both the old and new mapping of InterBase major error codes to SQLCODE numbers. *Appendix A* in the *Programmer's Reference*, contains a list of error messages, explanations of them, and possible corrective actions for them. The error messages listed there correspond to the **gds** Error Symbols listed below.

Table 5. Mapping of SQLCODE Numbers to InterBase Major Errors

Error Description	gds Error Symbol	Old SQLCODE Mapping	New SQLCODE Mapping
Invalid BLR request	gds_$invalid_blr	-901	-104
Attempted invalid blob operation	gds_$segstr_no_op	-901	-402
Attempted to write to a read-only blob	gds_$segstr_no_write	-901	-817
Metadata is obsolete	gds_$obsolete_metadata	-902	-820
Unsupported on-disk structure	gds_$wrong_ods	-901	-820
Implementation limit exceeded	gds_$imp_exc	-901	-904
Deadlock	gds_$deadlock	-912	-913
Bad parameters on attach or create database	gds_$bad_dpb_content	-901	-924
No match for first value expression	gds_$from_no_match	-901	+100
Invalid database key	gds_$no_record	-901	+100
Attempted to fetch past the last record in a record stream	gds_$stream_eof	-901	+100

Mapping of Minor Codes

The following table lists SQLCODE numbers which correspond to InterBase minor errors. Many InterBase minor errors are mapped to a single SQLCODE number. Where possible, check the SQLCODE number associated with the InterBase major code before acting on the error returned for the minor code.

Table 6. Mapping of SQLCODE Numbers to InterBase Minor Errors

Error Description	gds Error Symbol	SQLCODE Number
BLR syntax error	gds_$syntaxerr	-104
Context already in use (BLR error)	gds_$ctxinuse	-104
Context not defined (BLR error)	gds_$ctxnotdef	-104
Bad parameter number	gds_$badparnum	-104
Bad message vector	gds_$bad_msg_vec	-104
Invalid slice description language	gds_$invalid_sdl	-104
Cannot update	gds_$non_updatable	-150
Function could not be matched	gds_$funmismat	-171
Field not array or invalid dimensions	gds_$invalid_dimension	-171
Function not defined	gds_$funnotdef	-172
Generator not defined	gds_$gennotdef	-204
Field not defined in relation	gds_$fldnotdef	-205
Relation not defined	gds_$relnotdef	-219
Blob and array data types not supported for operation	gds_$blobnotsup	-402
Data operation not supported	gds_$datnotsup	-402
Subscript out of bounds	gds_$out_of_bounds	-406
Null segment of unique key	gds_$nullsegkey	-407
Filter not found to convert type	gds_$nofilter	-413

Table 6. Mapping of SQLCODE Numbers to InterBase Minor Errors

Error Description	gds Error Symbol	SQLCODE Number
Foreign key does not exist in primary index	gds_$foreign_key	-530
Primary key exists in foreign index	gds_$primary_key	-532
Key size larger than allowed for index	gds_$keytoobig	-664
Extension error	gds_$ext_err	-677
Invalid blob type for operation	gds_$bad_segstr_type	-685
Operation not supported	gds_$read_only	-817
Wrong DYN version	gds_$wrodynver	-820
Minor version too high	gds_$high_minor	-820
Internal error	gds_$bdbincon	-901
Database handle not zero	gds_$dbbnotzer	-901
Transaction handle not zero	gds_$tranotzer	-901
Transaction in limbo	gds_$trainlim	-901
Transaction not in limbo	gds_$notinlim	-901
Transaction outstanding	gds_$traoutsta	-901
Undefined message number	gds_$badmsgnum	-901
Blocking signal has been received	gds_$blocking_signal	-901
Database system cannot read argument	gds_$noargacc_read	-901
Database system cannot write argument	gds_$noargacc_write	-901
Internal error	gds_$misc_interpreted	-901
Transaction status error	gds_$tra_state	-901
Internal error	gds_$badblk	-902
Internal error	gds_$invpoolc	-902
Internal error	gds_$relbadblk	-902

Table 6. Mapping of SQLCODE Numbers to InterBase Minor Errors

Error Description	gds Error Symbol	SQLCODE Number
Block size larger than allowed	gds_$blktoobig	-902
Incompatible version of on-disk structure	gds_$badodsver	-902
Internal error	gds_$dirtypage	-902
Internal error	gds_$waifortra	-902
Internal error	gds_$doubleloc	-902
Internal error	gds_$nodnotfnd	-902
Internal error	gds_$dupnodfnd	-902
Internal error	gds_$locnotmar	-902
Page is of wrong type	gds_$badpagtyp	-902
Database corrupted	gds_$corrupt	-902
Checksum error on database page	gds_$badpage	-902
Index is broken	gds_$badindex	-902
Transaction synchronization error	gds_$trareqmis	-902
Bad handle count	gds_$badhndcnt	-902
Wrong version of transaction parameter block	gds_$wrotpbver	-902
Unsupported BLR version	gds_$wroblrver	-902
Wrong version of database parameter block	gds_$wrodpbver	-902
Database corrupted	gds_$badrelation	-902
Internal error	gds_$nodetach	-902
Internal error	gds_$notremote	-902
Internal error	gds_$dbfile	-902
Internal error	gds_$orphan	-902
Lock manager error	gds_$lockmanerr	-902

Table 6. Mapping of SQLCODE Numbers to InterBase Minor Errors

Error Description	gds Error Symbol	SQLCODE Number
SQL error code	gds_$sqlerr	-902
Bad sector information	gds_$bad_sec_info	-902
Invalid sector information	gds_$invalid_sec_info	-902
Too many requests	gds_$nopoolids	-904
Buffer exhausted	gds_$bufexh	-904
Buffer in use	gds_$bufinuse	-904
Request in use	gds_$reqinuse	-904
No lock manager available	gds_$no_lock_mgr	-904
Virtual memory exhausted	gds_$virmemexh	-904
Update conflicts with concurrent update	gds_$update_conflict	-904
Object in use	gds_$obj_in_use	-904
Cannot attach active shadow file	gds_$shadow_accessed	-904
File in manual shadow unavailable	gds_$shadow_missing	-904
Product not licensed	gds_$unlicensed	-906
Record from transaction stuck in limbo	gds_$rec_in_limbo	-911
Connection rejected by remote interface	gds_$connect_reject	-923
Secondary server attachments cannot validate databases	gds_$cant_validate	-923
Secondary server attachments cannot start journaling	gds_$cant_start_journal	-923
Secondary server attachments cannot start logging	gds_$cant_start_logging	-923
Communication error with journal	gds_$journerr	-924
Database detach completed with errors	gds_$bad_detach	-924
No rollback performed	gds_$no_rollback	-926

SQL Set Operations

In InterBase V3.3, handling of SQL set operations (updates, inserts, and deletes) has been brought into tighter compliance with the ANSI standards. When a statement performs a set operation, if the specified operation fails for any qualifying record in the set, the entire operation is rolled back.

For example, if a statement attempts to insert records from one table into another table with unique index values, inserts will succeed only so long as the inserted records do not duplicate index values. If an insert fails for any reason, *all* insertions performed by that statement will be rolled back.

qli Record and Set Behavior

In InterBase V3.3, the way **qli** handles loop operations is modified. Previously, operations performed on a relation using a **for** loop would continue until the operation failed for a single record, or until all records were successfully processed. In the case of a record-level failure, changes made to the relation up to the point of failure would remain intact.

In InterBase V3.3 there are now two possible scenarios: the previous record-level behavior, where successfully completed record changes are left intact; or complete **for** loop operation rollback. Which action **qli** takes depends on the statements in the **for** loop.

Record-Level Behavior

If the statement in a **qli for** loop both produces screen output and makes changes to a relation, then when the loop terminates prematurely, changes made to the relation up to the point of failure remain intact. This behavior is the same as record-level behavior in previous versions of InterBase.

For example, suppose you want to modify the city and state names for all records in the SKI_AREAS relation. Suppose, too, however, that a trigger is defined which prevents you from changing any records with a state name of "NH". To see the current value of each record you want to modify before you change it, you can use the **print then modify** construction:

```
QLI> for ski_areas print then modify city, state
```

This loop displays each record on your screen, then prompts you to enter a new city and state. As long as the state name is not "NH", you can change records. When you attempt to change a record with a state name of "NH", however, the trigger is activated, an error is reported to you, and the **for** loop is terminated. Because you have seen each record as it is processed, know where failure occurred and can therefore decide what action to take, changes you have made up to the point of failure are stored in the database. To abort the entire transaction, you must manually execute a **rollback** statement at the **qli** prompt.

Set Behavior

On the other hand, if you execute a **for** loop which *does not* produce screen output, **qli** treats the entire loop as a set operation. In the event that a change to a single record in the set fails, **qli** rolls back *all* changes in the **for** loop. This set behavior is new to InterBase V3.3.

For example, suppose you want to modify the state name to "CO" for all records in the SKI_AREAS relation. To change the records you execute the following **qli** statement:

```
QLI> for ski_areas modify using state = "CO"
```

Suppose, too, however, that a trigger is defined which prevents you from changing any records with a state name of "NH". In this case, **qli** begins processing records until the trigger is fired. You receive an error message, and since you do not know which record caused the **for** loop to terminate, **qli** rolls back *all* **for** loop changes.

Backing up to and Restoring from Tape

The capability of backing up to and restoring from multiple sources (multiple cartridge tapes or floppy disks, or multiple devices) has been added to **gbak** in V3.3. When a single tape or floppy disk has been filled during backup or all the data has been read from the tape or floppy disk during a restore, the following message is displayed:

```
Done with volume # <vol_num>, "<device_name>"
    Press return to reopen that file, or type a new
    name followed by return to open a different file.
Name:<default_device_name>
```

The volume number and default device name are displayed. The user inserts the next tape or floppy disk and continues until the backup or restore is complete. To backup to or restore from a different device, the user can enter a new device name rather than use the default.

gbak Switch

Previously, description and source information from RDB$_DESCRIPTION and RDB$_SOURCE was not formatted correctly when restoring a database using **gbak**. Line breaks did not appear in the correct place. This problem has been corrected in V3.3. Databases backed up and restored using the V3.3 **gbak** will be restored correctly.

However, a new **gbak** switch has been added to V3.3 to insure backward compatibility. If you intend to restore a database using a previous version **gbak** that you have backed up using the V3.3 **gbak**, you must use this switch, **ol [old_descriptions]**, when you back up the database. If you back up a database in V3.3 without the switch and try to restore it using a previous version of **gbak**, the restore will fail.

gdef and qli Option

The **modify index** statement for **gdef** and **qli** has been enhanced by the addition of a new option, **statistics**. This option recomputes the selectivity of a specified index, and stores the resulting value in the new RDB$STATISTICS field in the RDB$INDICES system relation.

Index selectivity is used internally by the optimizer when calculating optimal query join order. To improve InterBase performance, recompute index selectivity often for databases which contain frequently modified data.

The syntax for using the **statistics** option is:

```
modify index <indexname> statistics
```

where <*indexname*> is the exact name of the index for which to recompute selectivity.

Unlike other **modify index** options, using the **statistics** option does *not* rebuild an index.

Lock Manager Semaphores

On Unix and Sun systems, the maximum number of semaphores that may be reserved for InterBase locking has been increased from 32 to 128. InterBase default settings are considerably lower and vary according to your system. To increase the number of reserved semaphores, you can modify the SEMCOUNT parameter in */usr/interbase/lock_header*. Instructions for creating and editing *lock_header* may be found in your InterBase installation guide.

Note

Do *not* modify SEMCOUNT without reading the following considerations.

Available Semaphores

The total number of semaphores available for *all* processes on your system is hard-coded in the operating system kernel, and may be less than 128. To increase the number of semaphores available for InterBase locking *and* all other processes on your system, you may have to reconfigure and rebuild your kernel. The following kernel configuration options are related to the SEMCOUNT parameter and may need to be modified. Some systems do not use all of these options, and some systems may not permit the total number of semaphores to be set as high as 128. Consult your Unix system documentation before making any kernel changes.

Table 7. Kernel Configuration Options

Option	Purpose
SEMMNS	Sets the maximum total number of semaphores for your system.
SEMMNI	Specifies the maximum number of semaphore clusters (groups of semaphores).
SEMMSL	Indicates the maximum number of semaphores per cluster.
SEMMNU	Sets the number of undo structures.

InterBase 3.2 Considerations

Applications linked with the *gds_b.a* back end shipped with InterBase V3.2 cannot take advantage of an increase to the number of lock semaphores. These applications should either be relinked with the *gds_b.a* back end shipped with V3.3, or, if possible, should be relinked to use shared libraries instead.

In an environment running both InterBase V3.2 and V3.3 applications, if the lock table is accessed first from a V3.3 database and the number of semaphores is greater than 32, then subsequent V3.2 database users will not be able to access the V3.3 database. If the lock table is accessed first from a V3.2 database, then any subsequent V3.3 database users will be limited to using a 32 semaphore lock table.

Events Manager Parameter

You can now extend the events manager parameter to indicate the amount of shared memory that can be used by events. InterBase currently uses the default value of 32768 bytes of shared memory for events. The new parameter, EVENT_SHMSIZE, uses a default value

and has been commented out. To increase the amount of shared memory used for events to greater than the default value, remove the comment characters from EVENT_SHMSIZE and type in the new value in bytes.

International Support for InterBase

Note

Most of the international features decribed below are available in the first wave of the V3.3 release; however, subtypes 139 through 152 and 154 do not appear in this wave. These subtypes, along with the corresponding subtype charts, will be provided in an upcoming update in the near future. Contact InterBase customer support to determine when these features will be available.

InterBase V3.3 includes capabilities for sharing data between different host language versions. It supports single-byte, eight bit data needs and national conventions for manipulation of text data.

Overview

Textual data is stored in text and varying fields within a database. Fields containing international text are identified by a numbered subtype definition. The subtype determines the conventions InterBase uses to manipulate the data. The subtype is a component of the global field definition. Relations using the field inherit the attributes of the global field. Each text subtype is implemented via an InterBase Language Driver. On operating systems that use shared libraries, these modules are installed in */usr/interbase/lib* or */interbase/lib*, depending upon your platform, and define functions that InterBase uses for text manipulation of international data. On non-shared library systems, the subtype modules are included in the normal InterBase libraries.

Data Definition

Fields that contain international data are defined by the pairing of a character set and a collation sequence, specified as a subtype. The pairing is referred to as the *interpretation* of the character type. The interpretation of a character type defines the range comparison, string search, and case conversion rules by which InterBase operates on the data.

V3.3 provides the following subtypes. The ISO Latin I character set is equivalent to the ANSI character set with points 0-31 and 127-159 undefined. Refer to the appendixes for

charts on DOS code pages 437, 865, and the Latin-1 character sets. Charts displaying sort order for each subtype are also included.

Table 8. Subtypes

Subtype	Character Set	Collation Sequence	Description
sub_type 0 (default)	ASCII	ASCII	Rules for collation follow United States standards. Character codes with the 8th-bit set are undefined.
sub_type 1 (fixed)	none (binary data)	none (collates binary order)	InterBase "fixed" data type. Character set is considered to be binary. Values are null padded for comparison and collation.
sub_type 101	DOS CODE PAGE 437	Standard ASCII	Sort order compatible with Paradox 4.0 for DOS. Standard United States ASCII sort order.
sub_type 102	DOS CODE PAGE 437	International	Sort order compatible with Paradox 4.0 for DOS. International sort order.
sub_type 105	DOS CODE PAGE 865	Nordan 4	Sort order compatible with Paradox 4.0 for DOS. Nordan 4 sort order.
sub_type 106	DOS CODE PAGE 437	SwedFin	Sort order compatible with Paradox 4.0 for DOS. SwedFin sort order.
sub_type 139	ISO8859.1 (Latin-1)	Denmark	Latin-1 text using Danish rules for collation and uppercasing.
sub_type 140	ISO8859.1 (Latin-1)	Netherlands	Latin-1 text using Dutch rules for collation and uppercasing.
sub_type 141	ISO8859.1 (Latin-1)	Finland	Latin-1 text using Finnish rules for collation and uppercasing.
sub_type 142	ISO8859.1 (Latin-1)	France	Latin-1 text using French rules for collation and uppercasing.
sub_type 143	ISO8859.1 (Latin-1)	FrenchCan	Latin-1 text using French Canadian rules for collation and uppercasing.
sub_type 144	ISO8859.1 (Latin-1)	Germany	Latin-1 text using German rules for collation and uppercasing.

Table 8. Subtypes

Subtype	Character Set	Collation Sequence	Description
sub_type 145	ISO8859.1 (Latin-1)	Iceland	Latin-1 text using Icelandic rules for collation and uppercasing.
sub_type 146	ISO8859.1 (Latin-1)	Italy	Latin-1 text using Italian rules for collation and uppercasing.
sub_type 149	ISO8859.1 (Latin-1)	Spain	Latin-1 text using Spanish rules for collation and uppercasing.
sub_type 151	ISO8859.1 (Latin-1)	Sweden	Latin-1 text using Swedish rules for collation and uppercasing.
sub_type 152	ISO8859.1 (Latin-1)	UK	Latin-1 text using British rules for collation and uppercasing.
sub_type 153	ISO8859.1 (Latin-1)	US	Latin-1 text using United States rules for collation and uppercasing. Character codes with the 8th-bit set are defined.
sub_type 154	ISO8859.1 (Latin-1)	Portugal	Latin-1 text using Portuguese rules for collation and uppercasing.

In the following GDML example, a sample database is defined using subtype 144 (German Latin-1) and subtype 142 (French Latin-1):

```
define database ''subtype_test'';
/*Global Field Definitions*/
define field ENGLISH_NAME char [80];
define field FRENCH_NAME char[30] sub_type 142;
define field GERMAN_NAME char[40] sub_type 144;
/*Relation Definition*/
define relation NOUNS
    ENGLISH_NAME position 0,
    FRENCH_NAME position 0,
    GERMAN_NAME position 2;
```

InterBase will now use culturally correct rules for French when manipulating FRENCH_NAME. Indexes defined using GDML will use the collation order defined by the subtype associated with the global field.

The example below demonstrates table creation using SQL:

```
create table nouns (
    ENGLISH_NAME char (30),
    GERMAN_NAME char (40),
    FRENCH_NAME char (30));
```

Since SQL does not contain operators for defining international data, the subtype must be set by directly modifying the RDB$FIELDS system relation:

```
commit;

update RDB$FIELDS set
    RDB$FIELD_SUB_TYPE = 142 where
RDB$FIELD_NAME = (
    select rf.RDB$FIELD_SOURCE from
           RDB$RELATION_FIELDS rf where
    rf.RDB$RELATION_NAME = ''NOUNS'' and
    rf.RDB$FIELD_NAME = ''FRENCH_NAME''
    )
        .
        .
        .
```

Indexes defined using SQL will use the collation order defined by the subtype associated with the data column. For example:

```
create index NOUNS_IDX1 on NOUNS ( FRENCH_NAME );
```

The following rules apply to data definition:

- Fields containing international data are defined globally and must be identified by a numbered subtype definition.
- Relations using a field with an associated subtype inherit the attributes of the global field.
- Subtypes 0 and 1 are reserved for ASCII and binary-byte fields. All character fields within system relations are of subtype 0.
- There can be more than one type of international text represented in a database, each subtype defines the rules for that type of text, not for the database as a whole. Relations can contain fields of different subtypes.
- Comparisons between different interpretations (subtypes) return results which are undefined. In V3.3, no error will be returned and the results from such comparisons will follow the rules for the highest numbered interpretation (subtype).
- Comparisons between different interpretations will not be allowed in future InterBase versions. A casting operator will be implemented and an explicit cast between text of different types will be required.
- Subtypes should be available in */usr/interbase/lib* or */interbase/lib* prior to database definition. In V3.3, no error will be generated when a global field is defined using an undefined subtype. In future InterBase versions, using an undefined subtype will generate an error.
- Use only global fields that have valid subtypes when defining a relation. Although a global field may be defined for a subtype that does not exist, using the field in a relation definition will produce an error.
- Operators like concatenation, which return strings, should be performed between strings of the same interpretation. The subtype of a concatenated expression is the same as the first operand.
- If the subtype associated with a global field is redefined, the associated indexes must be rebuilt and reactivated.

Operators

The rules by which InterBase operates on international data are derived from the subtype associated with the global field. The examples below indicate some of these operations:

Table 9. Operations Examples

Operation	Data Lang	Example	Comment
Collation	GDML	define index FRENCH_ORDER for NOUNS FRENCH_NAME;	The rules for indexing are inherited from the text subtype of FRENCH_NAME
	SQL	create index FRENCH_ORDER on NOUNS (FRENCH_NAME);	
Sorting	GDML	for NOUNS sorted by FRENCH_NAME print;	Prints the NOUNS relation sorted by the rules for text subtype FRENCH_NAME. If an index exists for FRENCH_NAME, retrieval follows that order.
	SQL	select * from NOUNS order by FRENCH_NAME;	
Comparison	GDML	for NOUNS with ENGLISH_NAME > "milk" print;	The comparison between ENGLISH_NAME and "milk" follows the rules for subtype 0, in this case ASCII.
	SQL	select * from NOUNS where ENGLISH_NAME >"milk";	
Between	GDML	for NOUNS with ENGLISH_NAME between "fish" and "milk";	Between is equivalent to: ENGLISH_NAME >="fish" and ENGLISH_NAME<= "milk" and follows the comparison rules defined for ENGLISH_NAME.
	SQL	select * from NOUNS where ENGLISH_NAME>= "fish" and ENGLISH_NAME<="milk";	
Upper Case	GDML	print NOUNS sorted by anycase FRENCH_NAME;	The uppercase rules for FRENCH_NAME are followed for the anycase operator.

Table 9. Operations Examples

Operation	Data Lang	Example	Comment
Joins	GDML	for NOUNS cross VERBS over FRENCH_NAME or for n in NOUNS cross v in VERBS with n.FRENCH_ NAME = v.FRENCH_NAME;	Follow rules for equality
Containing	GDML	print NOUNS with FRENCH_NAME containing "á";	The containing operator is case insensitive. It ignors case and follows the uppercase rules for the specified subtype.
Starting with	GDML	print NOUNS with FRENCH_NAME starting with "le";	**Starting with** follows character (byte) matching rules.
Matching	GDML	print NOUNS with ENGLISH_NAME matching "*ab*";	The **matching** operator follows character (byte) matching rules.

Accessing InterBase International Data With DOS

InterBase V3.3 international data can be accessed and manipulated from IBM PC-compatible machines running MS-DOS with appropriate software, like Borland's Paradox 4.0 with SQL Link for InterBase. Because the UNIX and DOS environments use different character sets, however, data passed from one environment to the other must be converted for display and, depending on need, for storage as well. Options for display, storage, and update conversion can involve one or more of the following tools:

- Views.
- User-defined functions (UDFs).
- Triggers.

All conversions must take place on the InterBase server side.

Sample Conversion Support Provided with InterBase

During installation of InterBase V3.3 a directory, */usr/interbase/intl*, is created to hold international examples, databases, a C code module which contains international UDFs, and header files containing conversion data tables. The main purpose of these files is to provide a working example of data conversion across platforms, but the UDF source code and conversion header files can also be adapted for user programs.

Note

Files provided in the international directory are unsupported. Source code is included to allow users to modify and adapt to their own programming needs. These files are not covered by the Support Agreements between Borland and its InterBase customers.

The following table lists the files in */usr/interbase/intl*.

Table 10: Files in /usr/interbase/intl

File	Description
cs_convert.c	C code module for the international UDFs
cs_demo.gdl	Database, UDF, and trigger data definition file
cs_load.gdl	Data definition file used to create a sample database, *cs_load.gdb*
cs_load.qli	Program to populate relations in *cs_load.gdb*
make.cs	Platform-specific makefile to compile and link international UDFs, and create the *cs_demo* database
437_to_865.h	C header file containing a data conversion table
437_to_lat1.h	C header file containing a data conversion table
865_to_lat1.h	C header file containing a data conversion table
clients.437	Data for the CLIENTS relation in *cs_demo.gdb*
contacts.437	Data for the CONTRACTS relation in *cs_demo.gdb*
products.lat1	Data for the PRODUCTS relation in *cs_demo.gdb*
users.lat1	Data for the USERS relation in *cs_demo.gdb*

Creating the International Examples

The */usr/interbase/intl* directory contains a database makefile, *make.cs,* which generates the sample international database, *cs_demo.gdb,* which contains definitions of the UDFs in *cs_convert.c.* To run *make.cs,* follow these steps:

1. Create a new directory to hold the example database.
2. Copy *make.cs* to the newly-created directory.
3. From the new directory, run these commands in order:

```
make -f make.cs
make -f make.cs new
make -f make.cs demo
```

The final command in this sequence creates a UDF library, CONVERT.

Contents of the cs_demo.gdb Database

The international demonstration database, *cs_demo.gdb,* created using the *make.cs* file described above, contains the following tables:

Table 11: List of Tables in *cs_demo.gdb*

Table	Collation Subtypes	How to Use with the Example Database
CLIENTS	102	Corresponds to DOS code page 437. Collation subtype 102 is the standard international sort order on IBM PC-compatibles. Can be inserted, updated, and viewed directly from DOS. Must be converted for proper display in UNIX. UNIX applications inserting or updating to this table must convert Latin 1 data to DOS code page 437 on database write.
CONTACTS	102	DOS code page 437. Collation subtype 102 is the standard international sort order on IBM PC-compatibles. Can be accessed directly from IBM PC-compatibles. Displayed as a computed field through a view, UNIX_CONTACTS, in UNIX.

Table 11: List of Tables in *cs_demo.gdb*

Table	Collation Subtypes	How to Use with the Example Database
PRODUCTS	102 and 153	DOS code page 437 and ISO8859.1 (Latin 1), the standard character set for most UNIX platforms. Subtype 153 is a standard United States collation and capitalization order. This table maintains two complete sets of data, one in DOS-accessible format, one in UNIX format. Applications on either platform can view their native formats directly. Inserts and updates made by an application in its native format must be applied collaterally, via GDML triggers, to the set of non-native data to keep both sets of data in synch.
USERS	153	ISO8859.1 (Latin 1). Subtype 153 is a standard United States collation and capitalization order. Can be inserted, updated, and viewed directly from UNIX. Must be converted for proper display in DOS. GDML triggers must be defined for this table, so that DOS applications inserting or updating to this table fire conversion routines to convert DOS code page 437 to Latin 1 on database write.

For a complete list of available collation subtypes, see *Data Definition* in *International Support for InterBase.*

The following table lists six views defined for the *cs_demo* database. These views convert stored data from one character set to another before display, but *do not* support character conversion for insertion and update.

Table 12: Views in *cs_demo.gdb*

View	Base Table	Purpose of View
PC_CONTACTS	CONTACTS	DOS client displays DOS code page 437 data.
PC_PRODUCTS	PRODUCTS	DOS client displays DOS code page 437 data from a table containing duplicate data for DOS code page 437 and ISO Latin 1.

Table 12: Views in *cs_demo.gdb*

View	Base Table	Purpose of View
PC_USERS	USERS	DOS client display requests are filtered through a UDF to translate ISO Latin 1 characters to DOS code page 437 characters on reads.
UNIX_CLIENTS	CLIENTS	UNIX client display requests are filtered through a UDF to translate DOS code page 437 characters to ISO Latin 1 characters on reads.
UNIX_CONTACTS	CONTACTS	UNIX client displays ISO Latin 1 data.
UNIX_PRODUCTS	PRODUCTS	UNIX client displays ISO Latin 1 data from a table containing duplicate data for ISO Latin 1 and DOS code page 437.

The following code fragment is the data definition for the UNIX_CLIENTS view:

```
define view UNIX_CLIENTS of c in CLIENTS
   CLIENT computed by (n_convert_437_to_Latin1(c.CLIENT, 32),
   c.COUNTRY
```

This view uses one of the sample UDFs supplied with InterBase V3.3, to perform character translation.

Note

Text translated from ISO Latin 1 to DOS code pages 437 and 865 may occasionally appear to contain oddly-placed blank spaces. DOS code pages 437 and 865 do not support some characters found in the ISO Latin 1 character set. During translation these characters are replaced with blanks.

Sample Conversion UDFs

Views which perform translation from one character set to another use sample international UDFs. For example, the UNIX_CLIENTS view uses a conversion UDF named **n_convert_437_to_Latin1()**, and PC_USERS uses a conversion UDF called **n_convert_Latin1_to_437()**. These international UDFs and several others, are coded in */usr/interbase/cs_convert.c*, and can be adapted as required for users' translation requirements. There are actually two complete sets of corresponding UDFs, one for use with null-terminated strings, and one for variable-length C strings which *do not* use terminating nulls.

The following table lists the conversion UDFs supplied in *cs_convert.c*.

Table 13: UDFs Supplied in *cs_convert.c*

Function Name	Function Name
convert_Latin1_to_437(<string>)	*n_convert_Latin1_to_437*(<string>, <length>)
convert_437_to_Latin1(<string>)	*n_convert_437_to_Latin1*(<string>, <length>)
convert_Latin1_to_865(<string>)	*n_convert_Latin1_to_865*(<string>, <length>)
convert_865_to_Latin1(<string>)	*n_convert_865_to_Latin1*(<string>, <length>)
convert_437_to_865(<string>)	*n_convert_437_to_865*(<string>, <length>)
convert_865_to_437(<string>)	*n_convert_865_to_437*(<string>, <length>)

Using the Example UDFs for Data Conversion

The character translation purposes for each example UDF listed in Table 4 is self-explanatory. There are, however, two translation routines provided for each possible character set translation. Which of these sets is appropriate depends on the kind of character strings requiring translation.

Use the **n_** functions when working with C strings (arrays of characters) which *do not* end with a null byte. The **n_** functions require two parameters, the address of the string to translate from one character set to another, and its length in bytes.

Use the other set of functions when working with strings which are demarcated by a terminating null byte. These functions require only the address of the string to translate from one character set to another.

Using Data Conversion Functions with Existing Data

To use the functions in *cs_convert.c* to convert existing data, embed them in C programs with GDML statements which access the data, or define database views and triggers which use the functions.

To use the UDFs in C programs with embedded GDML or SQL, the *cs_convert.c* module must be compiled and linked with the C program itself.

How Character Translation is Accomplished

The example conversion UDFs translate between character sets using data lookup tables. These tables are defined in the following *.h* files:

- */usr/interbase/intl/437_to_865.h*
- */usr/interbase/intl/437_to_lat1.h*
- */usr/interbase/intl/865_to_lat1.h*

During compilation, these files are automatically included in *cs_convert.c*. Users are free to examine and change these text files if desired, although current mappings are correct and as complete as possible. InterBase does not provide documentation or instructions for changing these files. Users who undertake changes do so at their own risk.

V3.3 Documentation Corrections

The InterBase V3.0 documentation was reprinted in June 1992 in the Borland format resulting in changes to the pagination from the original InterBase documentation set and the reprinted Borland documentation. The reprint version contains the same V3.0 functions with added corrections from *InterBase V3.2 Documentation Corrections* and *InterBase Previous Versions Documentation Corrections*. Consequently, there now exists two versions of essentially the same documentation: original documentation and reprinted documentation. The page numbers listed below indicate the changes:

(O), the page number in the original documentation set
(R), the page number in the reprinted documentation set
blank, the page number has not changed

Data Definition Guide

- On page 3-3, the database name should be in quotes when specifying a symbolic link:

```
% ln -s databases/atlas.gdb sample_db
GDEF> define database "sample_db";
```

- On pages (O) 5-7 and (R) 5-6, additional information should indicate that external relations cannot be defined using **qli**, only by using **gdef**.
- On pages (O) 8-17 and (R) 8-14, the first line of code is missing a semicolon:

```
modify database 'personnel.gdb';
```

DDL Reference

- On pages (O) 4-18 and (R) 4-14, the second paragraph is misleading. The **first** and **sorted** clauses *can* be used in triggers.
- On pages (O) 5-40 and (R) 5-33, the following should be added to the *computed-by-field* section: All computed fields should be computed by fields in the same relation and not across other relations.
- On pages (O) 5-44 and (R) 5-37, the UNIX line in the syntax box should read:

```
UNIX: grantee:==[gid,uid]
```

- On pages (O) 5-57 and (R) 5-49, the **order by** clause cannot be used in a view definition.

Database Operations

- On pages (O) 2-5 , (R) 2-4, and 6-7 the command to backup the sample database on a UNIX system onto a local cartridge or magnetic tape should be added:

```
% gbak -b atlas.gdb /dev/rmt8
```

- On page 6-2, the discussion of **d[ev] {mt | ct}** should mention that this function does *not* require the use of an option on UNIX platforms.
- On page 6-3, the section describing the **i[nactive]** option should read:

  Makes indexes inactive so you can restore a database with duplicate unique-index values. **gbak** does not work on databases with duplicate unique-index values unless you use this option. Use this option when you restore a database.

Programmer's Guide

- On page (O) 2-12, the last line of the second paragraph of code, and on page (R) 2-10, the last line of code should read:

```
based on badge_num.badge badge;
```

- On pages (O) 4-13 and (R) 4-11, at the beginning of the seventh line of code there is extra space that should be deleted.
- On pages (O) 5-8 and (R) 5-7, note that you cannot *access* a database via a UDF.
- On page 6-8, the fifth line of code should read:

```
for c in cities with c.city = cityname and c.state = statecode
```

- On pages (O) 8-20 and (R) 8-18, the **actual_seg_length** parameter in the arguments section for **gds_$get_segment** is an "out" not an "in".
- On pages (O) 8-21 and (R) 8-19, the third line of the arguments section (**actual_seg_length** ...) should be deleted.
- On page (R) 11-18, the third line from the bottom should read:

```
gds_$que_events (GDS_VAL(gds_$status), GDS_REF(DB),
```

- On pages (O) 13-12 and (R) 13-11, the second paragraph should read:

  When you finish the default transaction, close it with a **commit** or **rollback** command. However , if the last SQL query made was a metadata update (e.g., "create view"), an automatic **commit** is performed.

- On page 16-5, **alter table drop** *field* should not specify a data type. The example should read:

```
exec sql
    alter table ski_areas
        drop state
        add state char(2);
```

- On pages (O) 16-8 and (R) 16-7, the first example of defining a view in SQL has the line:

```
create view map cities as
```

it should be:

```
create view map_cities as
```

Programmer's Guide Omissions

- In chapter 15, the following example of using indicator variables on insert and update operations should be added:

```
/* the db definition:
define database "test.gdb";
define relation r1 x short, y short;
*/
database db = "test.gdb";
main()
{
short xvar, xind;
short yvar, yind;
ready;
start transaction;
isc_version(&db, (char *) 0, (char *) 0);
printf("Inserting record, x = 1, y = null\n");
xvar = 1;
xind = 0;
yvar = 0;
yind = -1
exec sql insert into r1 (x, y)
          values  (:xvar :xind,
                   :yvar :yind);
printf("Now selecting record\n");
xvar = 0;
yvar = 0;
```

```
xind = 0;
yind = 0;
exec sql select x, y into
            :xvar indicator :xind,
            :yvar indicator :yind
            from r1;
if (xind < 0)
            printf("x is missing: %d\t", xind);
else
            printf("x = %d\t", xvar);
if (yind < 0)
            printf("y is missing: %d\n", yind);
else
            printf("y = %d\n", yvar);
printf("Now trying update - setting x = null\n");
xvar = 0;
xind = -1;
yvar = 0;
yind = 0;
exec sql update r1 set x = :xvar indicator :xind where x = 1;
printf("Now reselecting records\n");
xvar = 0;
yvar = 0;
xind = 0;
yind = 0;
exec sql select x, y into
            :xvar indicator :xind,
            :yvar indicator :yind
            from r1;
if (xind < 0)
            printf("x is missing: %d\t", xind);
else
            printf("x = %d\t", xvar);
if (yind < 0)
            printf("y is missing: %d\n", yind);
else
            printf("y = %d\n", yvar);
rollback;
finish;
}
```

Programmer's Reference

- On page 2-4, the Reserved Words List should include USER.
- On page 3-6, the last line of the first paragraph says the test is case *in*sensitive. This is incorrect, the test *is* case sensitive.
- On pages (O) 4-46 and (R) 4-41, the *request option* paragraph should refer to the detailed description of its syntax and options found in the section beginning on pages (O) 4-91 and (R) 4-84.
- On page (O) 6-46 and (R) 6-39, the syntax for **execute immediate** should be:

```
execute immediate operation-name
```

qli Reference

- On pages (O) 3-66 and (R) 3-60, the letter "G" should be added to the table of *Date Edit String Characters* as follows: The character for exponentiation is "G". The default edit string for a FLOAT is G(8) and for a DOUBLE it is G(16). **qli** accepts the "G" character to alter the edit string.
- On pages (O) 3-134 and (R) 3-121, the following information should be added to the **column** *integer* option:

  When you do set columns in **qli** and run a procedure that prints to a file, the output will be in the columns' set width.

Previous Documentation Corrections

The following documentation corrections have been reported in previous releases in separate documents titled, *InterBase Previous Versions Documentation Corrections* and *InterBase V3.2 Documentation Corrections*. Most of these corrections were added to the June 1992 reprinted version of the V3.0 InterBase documentation set. However, the corrections listed below were not included in the reprint.

General Corrections

- When you back up a database that includes fields which are computed or validated through user defined functions (UDFs), you must have the UDF function library available when you restore the backup. During a restore, function libraries are needed to validate metadata.

If a required function library is not available, you will get an error message when you attempt to restore the data. If the source of the problem is not clear, rerun the **gbak** restore with the **-o** and **-v** options, which provide more information. A database recreated with the **-o** option includes an RDB$FUNCTIONS relation. RDB$FUNCTIONS contains the library name for each function.

- For VMS users, if you entered `"set exception break"` while in the debugger or have a `"set exception break"` line in your startup file, **gds_$attach_database** calls will cause execution to stop at exception breaks and you will see an error message about activating an image. These exception breaks are normal. You should type "go" after each break to continue execution.
- InterBase Version 2.*n* ignored transactions in limbo and used an older version of the record. Version 3.0 does not ignore limbo transactions; it reports an error if the process that initiated the limbo transaction is not live. If you do not want to see these error messages, use the **gds_$tpb_ignore_limbo** transaction option. This option does not use any arguments.
- On Apollo systems, **gds_$set_debug** forces the first process to attach to the database through the **gds_server** process instead of mapping the file directly. You define **gds_$set_debug** in a program as follows:

  ```
  std$call gds_$set_debug();
  ```

  To turn the feature on and force subsequent attachments to go through the server, use this call:

  ```
  gds_$set_debug( (long) 1);
  ```

  To turn the feature off so you can make subsequent attachments locally, use this call:

  ```
  gds_$set_debug( (long) 0);
  ```

  The argument for **gds_$set_debug** must be "long" and the parameter should be 0 or 1. Other values for the parameter may have unpredictable results.
- For Suns running OS 4.0 only, after you build a blob filter or a user defined function and copy the function library to */usr/interbase/lib*, you must run this command:

  ```
  ldconfig
  ```

  and then restart any servers that will use the filters or functions. For more information, see the chapters on blob filters and user defined functions in the *Programmer's Guide* and the *Data Definition Guide*.
- On all machines running Apollo SR10, you must compile blob filters and functions with the **-pic** option. On DN10000 machines, you must compile filters with the **-natural** option in addition to the **-pic** option. Refer to the *makefiles* in the */examples* directory for samples.

- Apollo systems are case-sensitive. The case in which you write C functions or filters must match the case you use when you call the function or filter.
- InterBase supports double quotation marks or single quotation marks (using the apostrophe key). For example, on pages O4-10 and N4-9 of *Getting Started with InterBase Guide*, the first SQL example should be either:

```
exec sql
    update rivers set length = 13
        where name = 'Jeffreys Creek';
if (SQLCODE)
    printf ("SQLCODE = %d\n", SQLCODE);
```

or

```
exec sql
    update rivers set length = 13
        where name = "Jeffreys Creek";
if (SQLCODE)
    printf ("SQLCODE = %d\n", SQLCODE);
```

- On operating systems with a fourteen character filename limitation, note the following changes to the names of the sample programs:

From	To
dsql_blob_display.e	*dsql_blob_disp.e*
dsql_blob_display.epas	*dsql_blob_disp.epas*
sql.multiple_db1.e	*sql_multiple_db1.e*
sql.multiple_db1.e	*sql_multiple_db2.e*

- The syntax for a UNIX grantee for the **define security_class** statement is shown below. The "groupid" and "userid" must be numbers.

```
UNIX: grantee:==[gid,uid]
```

- Changes to Forms (**fred**): If you are in a form, pressing the Tab key places the cursor on the next field, rather than the next editable field. Two form modification modes are available: navigation mode and edit mode. Use navigation mode to move from field to field; use edit mode to change modifiable fields. **fred** always starts in navigation mode, where you can use the arrow keys, Return key, Tab key, or Delete key to move from field to field. To shift into edit mode, move the cursor to a modifiable field and press the key that corresponds to Edit, Insert-Overstrike, Erase, or Insert (see the list below). To return to navigation mode, either press the Edit key or press any key that is not a field editing key (i.e., arrow, Tab, Return, or a function key).

Note

If you are using an HP machine and you are not using VT100 emulation mode, you must use the HOME key instead of the ENTER key.

Table 14: Forms Keys

Function	Description	Key On Apollo	Key On All Others
Edit	Toggles between edit mode and navigation mode	EDIT	Ctrl-G
Right	Moves cursor one character to the right	Right Arrow	Right Arrow
Left	Moves cursor one character to left	Left Arrow	Left Arrow
Delete	Deletes character to left of cursor	BACKSPACE	Delete
Delete-Next	Deletes current character	CHAR DEL	Ctrl-F
Go-To-Start	Moves cursor to first position of field	Left Bar Arrow	Ctrl-H
Go-To-End	Moves cursor to last position of field	Right Bar Arrow	Ctrl-E
Insert-Overstrike	Toggles between insert and overstrike modes	INS	Ctrl-A
Erase	Deletes contents of entire field	LINE DEL	Ctrl-U
Insert	Inserts any printable character into field	Any character	Any character

*Forms are not available on the HP 3000.

Using the Mouse: for Apollo workstations, you can use the mouse to navigate in forms and menus. The left mouse button corresponds to the ENTER key and the right button corresponds to the RETURN key. For horizontal and vertical menus, moving the mouse moves the menu cursor to the next or prior menu choice. You can also use the mouse to move the cursor among and within fields on a form.

- When an InterBase transaction does not change any data (i.e., read-only), you should commit that transaction rather than roll it back. The effect on the transaction is the same whether you commit or rollback, however the time and space required by subse-

quent transactions will be reduced if you commit. In the Version 3.0 documentation set, all examples that are read-only should commit rather than rollback.

- When you store or modify a blob, you cannot directly assign data to that blob. You must use this syntax:

```
<blob field name> = edit
```

  which opens the default editor, where you can enter or change the blob data.

- If you use the SQL **select** statement to retrieve data from a relation and the **group by** clause to group data together, the fields in the **group by** must uniquely select information from the **select** fields.

  For example, if you have the following fields, each of which contain the indicated data:

```
A   B   C   Valid Select/Group By
1   1   1   select A,B,C ... group by A,B,C
1   1   2   select A,B,max[C] ... group by A,B
2   3   1
2   3   2   Invalid Select/Group By
            select A,B,C ... group by A,B
```

- For languages, such as ADA and some versions of C, that do not support dollar signs ($) in identifiers, a single underscore is used instead of the underscore and dollar sign. For example, instead of PYXIS_$KEY_PF1, you would use:

```
PYXIS_KEY_PF1
```

- If you add a **valid_if** clause to a field and that field is in a relation with entries that do not meet the **valid_if** criteria, you will not be able to restore the database after using **gbak**.

To assure that **gds_$** function calls will work properly on all platforms, you should use the GDS_REF() macro with parameters passed by reference and the GDS_VAL() macro with parameters passed by value. For example:

```
gds_$ddl(gds_$status, GDS_REF(DB), GDS_REF(gds_$trans),
   sizeof(dyn_gdl), dyn_gdl;
```

GDS_REF(x) and GDS_VAL(x) are converted as follows:

	GDS_REF	GDS_VAL
on Apollo systems	x	*x
on non-Apollo systems	&x	x

Data Definition Guide

- On page O3-8 and O3-9, the examples that show how to define a shadow omit the keyword **manual** or **auto**. One of these keywords must be included in all **define shadow** statements. For example:

```
define shadow 1 auto 'atlas_shadow';
```

- On page O5-7, the external file specified in an external relation must reside on the same node as the primary database file with one exception: on VMS, a pathname containing a DECnet specification is permitted for the external file.
- On pages O9-16 and N9-15, additional information should indicate that user defined functions can be used in a **valid_if** clause.

Programmer's Guide

- Many of the examples in the *Programmer's Guide* and the *Programmer's Reference* state that when programming in the C language you should declare a transaction handle as a long integer. Although this works in most cases, you should declare a transaction handle as a long*, which is a pointer to a long integer.

Programmer's Reference

- On pages O6-19 and N6-17, the information for creating a view using embedded SQL is incorrect. Only the SQL source-language form of the definition is not stored in the database. As a result, you cannot extract the definition. The definition itself is permanently stored in the database in BLR form and the view may be used as if it were created in another language.
- On pages O6-31 and N6-28, the **delete** statement cannot be used with an alias.

qli Guide

- If you have readied more than one database, specified database handles, and want information for a specific database, you can use the database handle associated with that database with the **show** command:

```
show <db_handle.entity>
```

For example, if you assigned the handle "emp" and want to display the relations from the "emp" database, use this command:

```
show emp.relations
```

Bug Fixes

Bug#	Bug Fix Description
497	**qli** sub_type clause is now allowed in variable field definitions.
660	**gltj** prints out correct output when journalling is disabled.
1523	On Apollo, TCP/IP connections now allow access to secure relations.
3120	On AIX, MODULE_NAME now accepts the full path name rather than the 31-character limit.
3132	**gdef** now rejects **sort by** and **reduced to** in view definitions.
3219	References to array elements are now permitted in RSEs.
3414	Updating a relation with computed fields through a view no longer results in access violation.
3474	On VMS, **gfix -rollback** with a specific transaction now works.
3480	DSQL now supports unions of aggregates.
3488	**gstat** no longer results in an access violation when **qli print** is running.
3494	**commit** and **rollback** now close cursors.
3499	There is no longer a limit on the number of column entries that can be specified in a report.
3517	*perf.pas* now has the correct *perf* definition.
3528	**drop table** in one process no longer causes apparent corrupted database in another process.
3536	**gfix** now allows proper exit.
3547	**gpre** now generates correct code in **whenever** SQLWARNING.
3550	**-f** and **-n** switches are processed on VMS **gbak**, a message for **-n** has been added.
3576	FORTRAN programs that contain tabs now work correctly in **gpre**.
3588	Array elements can now be referenced in any **gdef** expression.
3601	Tables created in SQL can now have names longer than 27 characters, but the first 27 must be unique.
3612	**gds_$decode_date()** now returns the correct day of the week that corresponds to the date.
3617	**qli, jrd**, and DSQL now recognize the significance of a user supplied zero in selection criteria for a date field.
3619	See 3576.
3635	In DSQL, large values are correctly stored in a DOUBLE field.
3648	DSQL long integers now convert to floats correctly.
3649	DSQL allows aliases on insert.

3652	In DSQL, exponents may now be used as parameter markers.
3685	DSQL now accepts single quotation marks within a query.
3687	Correct error now returned from external relation consisting of an indexed file.
3704	**gdef** now reports syntax error on V2.5 trigger errors where **store, modify,** and **erase** were optional.
3705	See 3687.
3710	Multiple filter types are now determined correctly.
3712	**gbak** now preserves segments when backing-up and restoring the RDB$DESCRIPTION field. *See section describing new* ***gbak*** *switch.*
3713	Database key lengths for views with computed fields are now computed correctly.
3726	**set generator** *<generator_name>* **to** *<integer>* now works correctly.
3736	In AIX, the request synchronization error no longer occurs when triggers are used.
3737	Segmentation fault no longer occurs on **isc_transaction_info** when multiple databases are involved.
3747	SEMCOUNT has been increased to 128 semaphores. *See section on lock manager semaphores and the installation guides.*
3765	V3.3 XSQLDA corrects problem with SQLVAR length. *See section on the extended SQL descriptor area.*
3779	User can now access a view regardless of the permission granted on the base table if the base table and the view were created using **create table** and **create view** using **qli, gpre,** or DSQL. Using **define relation** in **gdef** will not work.
3780	DSQL rejects incorrectly phrased update **where current of** query.
3784	Invalid SQL **grant** statement no longer created by **gdef -e** option.
3786	**qli** now supports a count of up to one billion records.
3788	**isc_sql_interprete** is now correctly prototyped in *gds.hxx.*
3790	**gdef -e** now extracts SQL field-level security.
3792	On Apollo C++, EXTERN keyword now generates correct database handle. This no longer results in failure in multi-file applications.
3793	**gpre** now generates float arrays as statics rather than global fields.
3796	**gdef** properly extracts QUERY_HEADERS containing quotes.
3799	**gdef** allows orphaned triggers. You can now modify a view by dropping and redefining it without losing the triggers that were defined for the original view.

3801	DSQL programs on IMP which perform a **select** containing a greater than (>) comparison, no longer result in an error.
3804	On VMS, **gpre** now references the correct include file for C++.
3808	On AIX, an application no longer hangs in **gds_$cancel_events ()**.
3809	In **qli**, ^D no longer exits to shell.
3811	**set form** or **for form** in **qli** correctly displays message lines in **pyxis**.
3820	**gpre** maximum line length for ADA has been changed to 120 to accomodate ALSYS ADA restriction.
3831	On Motorola Delta, events now work through the pipe server.
3832	On Data General, event applications waiting on an event no longer cause problems.
3836	FORTRAN now uses "/" instead of EOF to indicate the end of input.
3837	ADA now uses "/" instead of EOF to indicate the end of input.
3839	Rounding of floating point numbers on SCO works correctly.
3845	**gbak** now correctly tests security class so that **grant** no longer fails.
3847	**gpre** correctly processes COBOL **perform** statements.
3851	An SQL query with an aggregate in a **grouped by** clause no longer causes a **qli** error.
3852	In DSQL, a query using ordinal position referenced in an **order by** clause no longer causes an error on Sun-4.
3853	**gpre** now gives the correct error message when an ordinal field position in an **order by** clause is out of range.
3854	**gpre** correctly handles COBOL line numbers.
3857	EVENT_SHMSIZE is now expandable. *See section on events manager parameter and the installation guides.*
3869	On Motorola IMP, summing expressions of fields by a **select** now works.
3876	Apollo installation script now has correct *inetd.conf*.
3890	DSQL now recognizes **avg**<*distinct field*>.
3891	On SGI, compiling *make.filter* no longer generates an error.
3892	Using FORTRAN on SUN, the OSRI calls **isc_attach_database, get_array_desc, isc_print_status, gen_id, gds_$trans,** and **isc_fetch** now work correctly.
3898	User defined function examples have been updated in */examples*.
3901	On Data General, FORTRAN **blob edit** is now consistent with other languages.
3927	COBOL parsing of periods terminating **start_transaction** statements has been corrected.

3928	**gpre** generates correct COBOL output.
3929	In **qli**, the second floating point overflow no longer causes an access violation.
3938	On AIX, **gfix -s** with a limbo transaction no longer results in an error.
3943	On Sun-4, a segmentation violation no longer occurs when SEMCOUNT> system MAX.
3946	The DYN string limitation no longer results in a segmentation fault if the input *.gdl* file limitation size is exceeded.
3950	Code is now generated for **release_requests** from COBOL.
3951	**gdef** no longer results in an error when processing *.gdl* file where a trigger on a view contains processing of that view.
3952	Retrieving records in sorted order, when an index exists to support that sort order, and adding records during the same transaction, now works correctly.
3958	On Sun and VMS, **gpre** no longer faults when detecting invalid SQL syntax.
3962	On VMS, **gpre** no longer splits declaration lines containing in-line comments that caused a computer error.
3968	In **qli**, the error message on **commit** statement has been corrected.
3975	DSQL no longer rejects associated parameters in an **in** clause.
3978	DSQL now sets SQLCODE to the proper value when creating a unique index and a duplicate value exists.
3979	An error will be reported if an index is created which is too long.
3991	Creating an SQL table with 31 characters in the name, when the name is unique in 27 characters, is now possible.
4000	Negative subtypes are now accepted when defining subtypes in DSQL using **create table** and **alter table**.
4005	**gpre** no longer allows an **order** clause on a view definition.
4026	On VMS, enabling journaling no longer reports a "journal file full" message on databases larger than 70,000 blocks.
4032	In **qli**, an "undetermined quote string" message no longer results when the first statement is an "or."
4043	**gds_$cancel_event** now works under **gds_cserver.**
4044	Defining a view using **union** no longer returns an error in DSQL.
4045	Creating a field based on an external database table that uses a user defined function in the validation clause no longer causes an access violation.
4048	Limbo transactions can now be rolled back without causing an error.

4051	Transposition of **qli** switch characters no longer results in continual display of error messages.
4056	Executing a **select** command with the **like** operator using the **escape** clause retrieves the correct records.
4067	Modifying and deleting a record within the same transaction no longer results in a "record too long" error.
4068	In DSQL, moving values from/to the SQLDA variables no longer results in a conversion error.
4080	On VMS, only **exit** will quit the installation procedure.
4081	On Sun, activating a shadow file with **gfix -a** and then attaching the original database using **qli,** no longer results in a segmentation violation.
4083	When passing a blob handle to or from a program, use **blr_qual** rather than **blr_blob**. Use the **blr_blob_id** declaration when you pass an open blob handle.
4094	Using a double in DSQL and embedded SQL no longer results in a conversion error.
4101	See 3836.
4110	Attempting to define an index on a view through DYN will produce an error.
4112	Attempting to drop a table for which permission has not been granted no longer results in **qli** exit.
4117	Processing of a fifteen-way reflexive join now works correctly.

Appendix A

Subtypes

Subtype	Character Set	Collation Sequence	Description
sub_type 0 (default)	ASCII	ASCII	Rules for collation follow United States standards. Character codes with the 8th-bit set are undefined.
sub_type 1 (fixed)	none (binary data)	none (collates binary order)	InterBase "fixed" datatype. Character set is considered to be binary. Values are null padded for comparison and collation.
sub_type 100	DOS CODE PAGE 437	ASCII dictionary sort***	Sort order compatible with Paradox 4.0 for DOS. Dictionary sort for A-Z.
sub_type 101	DOS CODE PAGE 437	Standard ASCII	Sort order compatible with Paradox 4.0 for DOS. Standard United States ASCII order.
sub_type 102	DOS CODE PAGE 437	International	Sort order compatible with Paradox 4.0 for DOS. International sort order.
sub_type 105	DOS CODE PAGE 865	Nordan	Compatible with Nordan sort order.
sub_type 106	DOS CODE PAGE 437	SwedFin	Compatible with Swedish/Finish sort order.
sub_type 139	ISO8859.1 (Latin-1)	Denmark	Latin-1 text using Danish rules for collation and uppercasing.
sub_type 140	ISO8859.1 (Latin-1)	Netherlands	Latin-1 text using Dutch rules for collation and uppercasing.
sub_type 141	ISO8859.1 (Latin-1)	Finland	Latin-1 text using Finish rules for collation and uppercasing.
sub_type 142	ISO8859.1 (Latin-1)	France	Latin-1 text using French rules for collation and uppercasing.

Subtypes

Subtype	Character Set	Collation Sequence	Description
sub_type 143	ISO8859.1 (Latin-1)	FrenchCan	Latin-1 text using French Canadian rules for collation and uppercasing.
sub_type 144	ISO8859.1 (Latin-1)	Germany	Latin-1 text using German rules for collation and uppercasing.
sub_type 145	ISO8859.1 (Latin-1)	Iceland	Latin-1 text using Icelandic rules for collation and uppercasing.
sub_type 146	ISO8859.1 (Latin-1)	Italy	Latin-1 text using Italian rules for collation and uppercasing.
sub_type 148	ISO8859.1 (Latin-1)	Norway	Latin-1 text using Norwegian rules for collation and uppercasing.
sub_type 149	ISO8859.1 (Latin-1)	Spain	Latin-1 text using Spanish rules for collation and uppercasing.
sub_type 151	ISO8859.1 (Latin-1)	Sweden	Latin-1 text using Swedish rules for collation and uppercasing.
sub_type 152	ISO8859.1 (Latin-1)	UK	Latin-1 text using British rules for collation and uppercasing.
sub_type 153	ISO8859.1 (Latin-1)	US	Latin-1 text using United States rules for collation and uppercasing. Character codes with the 8th-bit set are defined.
sub_type 154	ISO8859.1 (Latin-1)	Portugal	Latin-1 text using Portuguese rules for collation and uppercasing.

Appendix B

Table 15: DOS Code Page 437

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
0			32		Space
1	☺	White smiling face	33	!	Exclamation mark
2	☻	Black smiling face	34	"	Quotation mark
3	♥	Black heart suit	35	#	Number sign
4	♦	Black diamond suit	36	$	Dollar sign
5	♣	Black club suit	37	%	Percent sign
6	♠	Black spade suit	38	&	Ampersand
7	•	Bullet	39	'	Apostrophe-quote
8	◘	Inverse bullet	40	(	Opening parenthesis
9	○	White circle	41	)	Closing parenthesis
10	◙	Inverse white circle	42	*	Asterisk
11	♂	Male sign	43	+	Plus sign
12	♀	Female sign	44	,	Comma
13	♪	Eighth note	45	-	Hyphen-minus
14	♫	Barred eighth notes	46	.	Period
15	☼	White sun with rays	47	/	Slash
16	►	Black right pointing pointer	48	0	Digit zero
17	◄	Black left pointing pointer	49	1	Digit one
18	↕	Up down arrow	50	2	Digit two
19	‼	Double exclamation mark	51	3	Digit three
20	¶	Paragraph sign	52	4	Digit four
21	§	Section sign	53	5	Digit five
22	▬	Black rectangle	54	6	Digit six
23	↨	Up down arrow with base	55	7	Digit seven
24	↑	Up arrow	56	8	Digit eight
25	↓	Down arrow	57	9	Digit nine
26	→	Right arrow	58	:	Colon
27	←	Left arrow	59	;	Semicolon
28	∟	Right angle	60	<	Less-than sign
29	↔	Left right arrow	61	=	Equals sign
30	▲	Black up pointing triangle	62	>	Greater-than sign
31	▼	Black down pointing triangle	63	?	Question mark

Table 15: DOS Code Page 437

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
64	@	Commercial at	96	`	Spacing grave
65	A	Latin capital letter A	97	a	Latin small letter a
66	B	Latin capital letter B	98	b	Latin small letter b
67	C	Latin capital letter C	99	c	Latin small letter c
68	D	Latin capital letter D	100	d	Latin small letter d
69	E	Latin capital letter E	101	e	Latin small letter e
70	F	Latin capital letter F	102	f	Latin small letter f
71	G	Latin capital letter G	103	g	Latin small letter g
72	H	Latin capital letter H	104	h	Latin small letter h
73	I	Latin capital letter I	105	i	Latin small letter i
74	J	Latin capital letter J	106	j	Latin small letter j
75	K	Latin capital letter K	107	k	Latin small letter k
76	L	Latin capital letter L	108	l	Latin small letter l
77	M	Latin capital letter M	109	m	Latin small letter m
78	N	Latin capital letter N	110	n	Latin small letter n
79	O	Latin capital letter O	111	o	Latin small letter o
80	P	Latin capital letter P	112	p	Latin small letter p
81	Q	Latin capital letter Q	113	q	Latin small letter q
82	R	Latin capital letter R	114	r	Latin small letter r
83	S	Latin capital letter S	115	s	Latin small letter s
84	T	Latin capital letter T	116	t	Latin small letter t
85	U	Latin capital letter U	117	u	Latin small letter u
86	V	Latin capital letter V	118	v	Latin small letter v
87	W	Latin capital letter W	119	w	Latin small letter w
88	X	Latin capital letter X	120	x	Latin small letter x
89	Y	Latin capital letter Y	121	y	Latin small letter y
90	Z	Latin capital letter Z	122	z	Latin small letter z
91	[	Opening square bracket	123	{	Opening curly bracket
92	\	Backslash	124	|	Vertical bar
93	]	Closing square bracket	125	}	Closing curly bracket
94	^	Spacing circumflex	126	~	Tilde
95	_	Spacing underscore	127	⌂	House

Table 15: DOS Code Page 437

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
128	Ç	Latin capital letter C cedilla	160	á	Latin small letter a acute
129	ü	Latin small letter u diaeresis	161	í	Latin small letter i acute
130	é	Latin small letter e acute	162	ó	Latin small letter o acute
131	â	Latin small letter a circumflex	163	ú	Latin small letter u acute
132	ä	Latin small letter a diaeresis	164	ñ	Latin small letter n tilde
133	à	Latin small letter a grave	165	Ñ	Latin capital letter N tilde
134	å	Latin small letter a ring	166	ª	Feminine ordinal indicator
135	ç	Latin small letter c cedilla	167	º	Masculine ordinal indicator
136	ê	Latin small letter e circumflex	168	¿	Inverted question mark
137	ë	Latin small letter e diaeresis	169	⌐	Reversed not sign
138	è	Latin small letter e grave	170	¬	Not sign
139	ï	Latin small letter i diaeresis	171	½	Fraction one half
140	î	Latin small letter i circumflex	172	¼	Fraction one quarter
141	ì	Latin small letter i grave	173	¡	Inverted exclamation mark
142	Ä	Latin capital letter Adiaeresis	174	«	Left pointing guillemet
143	Å	Latin capital letter A ring	175	»	Right pointing guillemet
144	É	Latin capital letter E acute	176	░	Light shade
145	æ	Latin small letter ae	177	▒	Medium shade
146	Æ	Latin capital letter AE	178	▓	Dark shade
147	ô	Latin small letter o circumflex	179	│	Forms light vert
148	ö	Latin small letter o diaeresis	180	┤	Forms light vert and left
149	ò	Latin small letter o grave	181	╡	Forms vert sgl and left dbl
150	û	Latin small letter u circumflex	182	╢	Forms vert dbl and left sgl
151	ù	Latin small letter u grave	183	╖	Forms down dbl and left sgl
152	ÿ	Latin small letter y diaeresis	184	╕	Forms down sgl and left dbl
153	Ö	Latin capital letter O diaeresis	185	╣	Forms dbl vert and left
154	Ü	Latin capital letter U diaeresis	186	║	Forms dbl vert
155	¢	Cent sign	187	╗	Forms dbl down and left
156	£	Pound sign	188	╝	Forms dbl up and left
157	¥	Yen sign	189	╜	Forms up dbl and left sgl
158	₧	Peseta sign	190	╛	Forms up sgl and left dbl
159	ƒ	Latin small letter script f	191	┐	Forms light down and left

Table 15: DOS Code Page 437

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
192	└	Forms light up and right	224	α	Greek small letter alpha
193	┴	Forms light up and hor	225	ß	Latin small letter sharp s
194	┬	Forms light down and hor	226	Γ	Greek capital letter gamma
195	├	Forms light vert and right	227	π	Greek small letter pi
196	─	Forms light hor	228	Σ	Greek capital letter sigma
197	┼	Forms light vert and hor	229	σ	Greek small letter sigma
198	╞	Forms vert sgl and right dbl	230	µ	Micro sign
199	╟	Forms vert dbl and right sgl	231	τ	Greek small letter tau
200	╚	Forms dbl up and right	232	Φ	Greek capital letter phi
201	╔	Forms dbl down and right	233	Θ	Greek capital letter theta
202	╩	Forms dbl up and hor	234	Ω	Greek capital letter omega
203	╦	Forms dbl down and hor	235	δ	Greek small letter delta
204	╠	Forms dbl vert and right	236	∞	Infinity
205	═	Forms dbl hor	237	φ	Greek small letter phi
206	╬	Forms dbl vert and hor	238	ε	Greek small letter epsilon
207	╧	Forms up sgl and hor dbl	239	∩	Intersection
208	╨	Forms up dbl and hor sgl	240	≡	Identical to
209	╤	Forms down sgl and hor dbl	241	±	Plus-or-minus sign
210	╥	Forms down dbl and hor sgl	242	≥	Greater than or equal to
211	╙	Forms up dbl and right sgl	243	≤	Less than or equal to
212	╘	Forms up sgl and right dbl	244	⌠	Top half integral
213	╒	Forms down sgl and right dbl	245	⌡	Bottom half integral
214	╓	Forms down dbl and right sgl	246	÷	Division sign
215	╫	Forms vert dbl and hor sgl	247	≈	Almost equal to
216	╪	Forms vert sgl and hor dbl	248	°	Degree sign
217	┘	Forms light up and left	249	∙	Bullet operator
218	┌	Forms light down and right	250	·	Middle dot
219	■	Full block	251	√	Square root
220	▄	Lower half block	252	ⁿ	Superscript latin small letter n
221	▌	Left half block	253	²	Superscript digit two
222	▐	Right half block	254	■	Black square
223	▀	Upper half block	255		Non-breaking space

Table 16: DOS Code Page 865

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
0			32		Space
1	☺	White smiling face	33	!	Exclamation mark
2	☻	Black smiling face	34	"	Quotation mark
3	♥	Black heart suit	35	#	Number sign
4	♦	Black diamond suit	36	$	Dollar sign
5	♣	Black club suit	37	%	Percent sign
6	♠	Black spade suit	38	&	Ampersand
7	•	Bullet	39	'	Apostrophe-quote
8	◘	Inverse bullet	40	(	Opening parenthesis
9	○	White circle	41	)	Closing parenthesis
10	◙	Inverse white circle	42	*	Asterisk
11	♂	Male sign	43	+	Plus sign
12	♀	Female sign	44	,	Comma
13	♪	Eighth note	45	-	Hyphen-minus
14	♫	Barred eighth notes	46	.	Period
15	☼	White sun with rays	47	/	Slash
16	►	Black right pointing pointer	48	0	Digit zero
17	◄	Black left pointing pointer	49	1	Digit one
18	↕	Up down arrow	50	2	Digit two
19	‼	Double exclamation mark	51	3	Digit three
20	¶	Paragraph sign	52	4	Digit four
21	§	Section sign	53	5	Digit five
22	▬	Black rectangle	54	6	Digit six
23	↨	Up down arrow with base	55	7	Digit seven
24	↑	Up arrow	56	8	Digit eight
25	↓	Down arrow	57	9	Digit nine
26	→	Right arrow	58	:	Colon
27	←	Left arrow	59	;	Semicolon
28	∟	Right angle	60	<	Less-than sign
29	↔	Left right arrow	61	=	Equals sign
30	▲	Black up pointing triangle	62	>	Greater-than sign
31	▼	Black down pointing triangle	63	?	Question mark

Table 16: DOS Code Page 865

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
64	@	Commercial at	96	`	Spacing grave
65	A	Latin capital letter A	97	a	Latin small letter a
66	B	Latin capital letter B	98	b	Latin small letter b
67	C	Latin capital letter C	99	c	Latin small letter c
68	D	Latin capital letter D	100	d	Latin small letter d
69	E	Latin capital letter E	101	e	Latin small letter e
70	F	Latin capital letter F	102	f	Latin small letter f
71	G	Latin capital letter G	103	g	Latin small letter g
72	H	Latin capital letter H	104	h	Latin small letter h
73	I	Latin capital letter I	105	i	Latin small letter i
74	J	Latin capital letter J	106	j	Latin small letter j
75	K	Latin capital letter K	107	k	Latin small letter k
76	L	Latin capital letter L	108	l	Latin small letter l
77	M	Latin capital letter M	109	m	Latin small letter m
78	N	Latin capital letter N	110	n	Latin small letter n
79	O	Latin capital letter O	111	o	Latin small letter o
80	P	Latin capital letter P	112	p	Latin small letter p
81	Q	Latin capital letter Q	113	q	Latin small letter q
82	R	Latin capital letter R	114	r	Latin small letter r
83	S	Latin capital letter S	115	s	Latin small letter s
84	T	Latin capital letter T	116	t	Latin small letter t
85	U	Latin capital letter U	117	u	Latin small letter u
86	V	Latin capital letter V	118	v	Latin small letter v
87	W	Latin capital letter W	119	w	Latin small letter w
88	X	Latin capital letter X	120	x	Latin small letter x
89	Y	Latin capital letter Y	121	y	Latin small letter y
90	Z	Latin capital letter Z	122	z	Latin small letter z
91	[	Opening square bracket	123	{	Opening curly bracket
92	\	Backslash	124	|	Vertical bar
93	]	Closing square bracket	125	}	Closing curly bracket
94	^	Spacing circumflex	126	~	Tilde
95	_	Spacing underscore	127	⌂	House

Table 16: DOS Code Page 865

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
128	Ç	Latin capital letter C cedilla	160	á	Latin small letter a acute
129	ü	Latin small letter u diaeresis	161	í	Latin small letter i acute
130	é	Latin small letter e acute	162	ó	Latin small letter o acute
131	â	Latin small letter a circumflex	163	ú	Latin small letter u acute
132	ä	Latin small letter a diaeresis	164	ñ	Latin small letter n tilde
133	à	Latin small letter a grave	165	Ñ	Latin capital letter N tilde
134	å	Latin small letter a ring	166	ª	Feminine ordinal indicator
135	ç	Latin small letter c cedilla	167	º	Masculine ordinal indicator
136	ê	Latin small letter e circumflex	168	¿	Inverted question mark
137	ë	Latin small letter e diaeresis	169	⌐	Reversed not sign
138	è	Latin small letter e grave	170	¬	Not sign
139	ï	Latin small letter i diaeresis	171	½	Fraction one half
140	î	Latin small letter i circumflex	172	¼	Fraction one quarter
141	ì	Latin small letter i grave	173	¡	Inverted exclamation mark
142	Ä	Latin capital letter Adiaeresis	174	«	Left pointing guillemet
143	Å	Latin capital letter A ring	175	¤	Currency sign
144	É	Latin capital letter E acute	176	░	Light shade
145	æ	Latin small letter ae	177	▒	Medium shade
146	Æ	Latin capital letter AE	178	▓	Dark shade
147	ô	Latin small letter o circumflex	179	│	Forms light vert
148	ö	Latin small letter o diaeresis	180	┤	Forms light vert and left
149	ò	Latin small letter o grave	181	╡	Forms vert sgl and left dbl
150	û	Latin small letter u circumflex	182	╢	Forms vert dbl and left sgl
151	ù	Latin small letter u grave	183	╖	Forms down dbl and left sgl
152	ÿ	Latin small letter y diaeresis	184	╕	Forms down sgl and left dbl
153	Ö	Latin capital letter O diaeresis	185	╣	Forms dbl vert and left
154	Ü	Latin capital letter U diaeresis	186	║	Forms dbl vert
155	¢	Cent sign	187	╗	Forms dbl down and left
156	£	Pound sign	188	╝	Forms dbl up and left
157	Ø	Latin capital letter O slash	189	╜	Forms up dbl and left sgl
158	₧	Peseta sign	190	╛	Forms up sgl and left dbl
159	ƒ	Latin small letter script f	191	┐	Forms light down and left

Table 16: DOS Code Page 865

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
192	└	Forms light up and right	224	α	Greek small letter alpha
193	┴	Forms light up and hor	225	ß	Latin small letter sharp s
194	┬	Forms light down and hor	226	Γ	Greek capital letter gamma
195	├	Forms light vert and right	227	π	Greek small letter pi
196	─	Forms light hor	228	Σ	Greek capital letter sigma
197	┼	Forms light vert and hor	229	σ	Greek small letter sigma
198	╞	Forms vert sgl and right dbl	230	µ	Micro sign
199	╟	Forms vert dbl and right sgl	231	τ	Greek small letter tau
200	╚	Forms dbl up and right	232	Φ	Greek capital letter phi
201	╔	Forms dbl down and right	233	Θ	Greek capital letter theta
202	╩	Forms dbl up and hor	234	Ω	Greek capital letter omega
203	╦	Forms dbl down and hor	235	δ	Greek small letter delta
204	╠	Forms dbl vert and right	236	∞	Infinity
205	═	Forms dbl hor	237	φ	Greek small letter phi
206	╬	Forms dbl vert and hor	238	ε	Greek small letter epsilon
207	╧	Forms up sgl and hor dbl	239	∩	Intersection
208	╨	Forms up dbl and hor sgl	240	≡	Identical to
209	╤	Forms down sgl and hor dbl	241	±	Plus-or-minus sign
210	╥	Forms down dbl and hor sgl	242	≥	Greater than or equal to
211	╙	Forms up dbl and right sgl	243	≤	Less than or equal to
212	╘	Forms up sgl and right dbl	244	⌠	Top half integral
213	╒	Forms down sgl and right dbl	245	⌡	Bottom half integral
214	╓	Forms down dbl and right sgl	246	÷	Division sign
215	╫	Forms vert dbl and hor sgl	247	≈	Almost equal to
216	╪	Forms vert sgl and hor dbl	248	°	Degree sign
217	┘	Forms light up and left	249	•	Bullet operator
218	┌	Forms light down and right	250	·	Middle dot
219	▉	Full block	251	√	Square root
220	▄	Lower half block	252	ⁿ	Superscript latin small letter n
221	▌	Left half block	253	²	Superscript digit two
222	▐	Right half block	254	■	Black square
223	▀	Upper half block	255		Non-breaking space

Table 17: Code Page Latin-1

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
0			32		Space
1		Start of heading	33	!	Exclamation mark
2		Start of text	34	"	Quotation mark
3		End of text	35	#	Number sign
4		End of transmission	36	$	Dollar sign
5		Enquiry	37	%	Percent sign
6		Acknowledge	38	&	Ampersand
7		Bell	39	'	Apostrophe-quote
8		Backspace	40	(	Opening parenthesis
9		Horizontal tabulation	41	)	Closing parenthesis
10		Line feed	42	*	Asterisk
11		Vertical tabulation	43	+	Plus sign
12		Form feed	44	,	Comma
13		Carriage return	45	-	Hyphen-minus
14		Shift out	46	.	Period
15		Shift in	47	/	Slash
16		Data link escape	48	0	Digit zero
17		Device control one	49	1	Digit one
18		Device control two	50	2	Digit two
19		Device control three	51	3	Digit three
20		Device control four	52	4	Digit four
21		Negative acknowledge	53	5	Digit five
22		Synchronous idle	54	6	Digit six
23		End of transmission block	55	7	Digit seven
24		Cancel	56	8	Digit eight
25		End of medium	57	9	Digit nine
26		Substitute	58	:	Colon
27		Escape	59	;	Semicolon
28		File separator	60	<	Less-than sign
29		Group separator	61	=	Equals sign
30		Record separator	62	>	Greater-than sign
31		Unit separator	63	?	Question mark

Table 17: Code Page Latin-1

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
64	@	Commercial at	96	`	Spacing grave
65	A	Latin capital letter A	97	a	Latin small letter a
66	B	Latin capital letter B	98	b	Latin small letter b
67	C	Latin capital letter C	99	c	Latin small letter c
68	D	Latin capital letter D	100	d	Latin small letter d
69	E	Latin capital letter E	101	e	Latin small letter e
70	F	Latin capital letter F	102	f	Latin small letter f
71	G	Latin capital letter G	103	g	Latin small letter g
72	H	Latin capital letter H	104	h	Latin small letter h
73	I	Latin capital letter I	105	i	Latin small letter i
74	J	Latin capital letter J	106	j	Latin small letter j
75	K	Latin capital letter K	107	k	Latin small letter k
76	L	Latin capital letter L	108	l	Latin small letter l
77	M	Latin capital letter M	109	m	Latin small letter m
78	N	Latin capital letter N	110	n	Latin small letter n
79	O	Latin capital letter O	111	o	Latin small letter o
80	P	Latin capital letter P	112	p	Latin small letter p
81	Q	Latin capital letter Q	113	q	Latin small letter q
82	R	Latin capital letter R	114	r	Latin small letter r
83	S	Latin capital letter S	115	s	Latin small letter s
84	T	Latin capital letter T	116	t	Latin small letter t
85	U	Latin capital letter U	117	u	Latin small letter u
86	V	Latin capital letter V	118	v	Latin small letter v
87	W	Latin capital letter W	119	w	Latin small letter w
88	X	Latin capital letter X	120	x	Latin small letter x
89	Y	Latin capital letter Y	121	y	Latin small letter y
90	Z	Latin capital letter Z	122	z	Latin small letter z
91	[	Opening square bracket	123	{	Opening curly bracket
92	\	Backslash	124	|	Vertical bar
93	]	Closing square bracket	125	}	Closing curly bracket
94	^	Spacing circumflex	126	~	Tilde
95	_	Spacing underscore	127		Delete

Table 17: Code Page Latin-1

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
128	■	Undefined	160		Non-breaking space
129	■	Undefined	161	¡	Inverted exclamation mark
130	■	Undefined	162	¢	Cent sign
131	■	Undefined	163	£	Pound sign
132	■	Undefined	164	¤	Currency sign
133	■	Undefined	165	¥	Yen sign
134	■	Undefined	166	¦	Broken vertical bar
135	■	Undefined	167	§	Section sign
136	■	Undefined	168	¨	Spacing diaeresis
137	■	Undefined	169	©	Copyright sign
138	■	Undefined	170	ª	Feminine ordinal indicator
139	■	Undefined	171	«	Left pointing guillemet
140	■	Undefined	172	¬	Not sign
141	■	Undefined	173	-	Soft hyphen
142	■	Undefined	174	®	Registered trade mark sign
143	■	Undefined	175	¯	Spacing macron
144	■	Undefined	176	°	Degree sign
145	■	Undefined	177	±	Plus-or-minus sign
146	■	Undefined	178	²	Superscript digit two
147	■	Undefined	179	³	Superscript digit three
148	■	Undefined	180	´	Spacing acute
149	■	Undefined	181	µ	Micro sign
150	■	Undefined	182	¶	Paragraph sign
151	■	Undefined	183	·	Middle dot
152	■	Undefined	184	¸	Spacing cedilla
153	■	Undefined	185	¹	Superscript digit one
154	■	Undefined	186	º	Masculine ordinal indicator
155	■	Undefined	187	»	Right pointing guillemet
156	■	Undefined	188	¼	Fraction one quarter
157	■	Undefined	189	½	Fraction one half
158	■	Undefined	190	¾	Fraction three quarters
159	■	Undefined	191	¿	Inverted question mark

Table 17: Code Page Latin-1

CODE	CHAR	DESCRIPTION	CODE	CHAR	DESCRIPTION
192	À	Latin capital letter A grave	224	à	Latin small letter a grave
193	Á	Latin capital letter A acute	225	á	Latin small letter a acute
194	Â	Latin capital letter A circumflex	226	â	Latin small letter a circumflex
195	Ã	Latin capital letter A tilde	227	ã	Latin small letter a tilde
196	Ä	Latin capital letter A diaeresis	228	ä	Latin small letter a diaeresis
197	Å	Latin capital letter A ring	229	å	Latin small letter a ring
198	Æ	Latin capital letter AE	230	æ	Latin small letter ae
199	Ç	Latin capital letter C cedilla	231	ç	Latin small letter c cedilla
200	È	Latin capital letter E grave	232	è	Latin small letter e grave
201	É	Latin capital letter E acute	233	é	Latin small letter e acute
202	Ê	Latin capital letter E circumflex	234	ê	Latin small letter e circumflex
203	Ë	Latin capital letter E diaeresis	235	ë	Latin small letter e diaeresis
204	Ì	Latin capital letter I grave	236	ì	Latin small letter i grave
205	Í	Latin capital letter I acute	237	í	Latin small letter i acute
206	Î	Latin capital letter I circumflex	238	î	Latin small letter i circumflex
207	Ï	Latin capital letter I diaeresis	239	ï	Latin small letter i diaeresis
208	Ð	Latin capital letter Eth	240	ð	Latin small letter eth
209	Ñ	Latin capital letter N tilde	241	ñ	Latin small letter n tilde
210	Ò	Latin capital letter O grave	242	ò	Latin small letter o grave
211	Ó	Latin capital letter O acute	243	ó	Latin small letter o acute
212	Ô	Latin capital letter O circumflex	244	ô	Latin small letter o circumflex
213	Õ	Latin capital lette rO tilde	245	õ	Latin small letter o tilde
214	Ö	Latin capital letter O diaeresis	246	ö	Latin small letter o diaeresis
215	×	Multiplication sign	247	÷	Division sign
216	Ø	Latin capital letter O slash	248	ø	Latin small letter o slash
217	Ù	Latin capital letter U grave	249	ù	Latin small letter u grave
218	Ú	Latin capital letter U acute	250	ú	Latin small letter u acute
219	Û	Latin capital letter U circumflex	251	û	Latin small letter u circumflex
220	Ü	Latin capital letter U diaeresis	252	ü	Latin small letter u diaeresis
221	Ý	Latin capital letter Y acute	253	ý	Latin small letter y acute
222	Þ	Latin capital letter Thorn	254	þ	Latin small letter thorn
223	ß	Latin small letter sharp s	255	ÿ	Latin small letter y diaeresis

Appendix C

Subtype #101 Collation Sequence - ASCII

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	g	103	s	115	ë	137	ó	162
0	48	H	72	T	84	è	138	ú	163
1	49	h	104	t	116	ï	139	ñ	164
2	50	I	73	U	85	î	140	Ñ	165
3	51	i	105	u	117	ì	141	ª	166
4	52	J	74	V	86	Ä	142	º	167
5	53	j	106	v	118	Å	143	¿	168
6	54	K	75	W	87	É	144	α	224
7	55	k	107	w	119	æ	145	ß	225
8	56	L	76	X	88	Æ	146	Γ	226
9	57	l	108	x	120	ô	147	π	227
A	65	M	77	Y	89	ö	148	Σ	228
a	97	m	109	y	121	ò	149	σ	229
B	66	N	78	Z	90	û	150	µ	230
b	98	n	110	z	122	ù	151	τ	231
C	67	O	79	Ç	128	ÿ	152	Φ	232
c	99	o	111	ü	129	Ö	153	Θ	233
D	68	P	80	é	130	Ü	154	Ω	234
d	100	p	112	â	131	¢	155	δ	235
E	69	Q	81	ä	132	£	156	∞	236
e	101	q	113	à	133	¥	157	∅	237
F	70	R	82	å	134	ƒ	159	ε	238
f	102	r	114	ç	135	á	160	∩	239
G	71	S	83	ê	136	í	161	n	252

DOS Code Page 437

Subtype #102 Collation Sequence - International

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	d	100	L	76	t	116	¢	155
0	48	D	68	m	109	T	84	£	156
1	49	e	101	M	77	u	117	¥	157
2	50	ë	137	n	110	ú	163	*f*	159
3	51	é	130	ñ	164	ù	151	ª	166
4	52	è	138	N	78	û	150	º	167
5	53	ê	136	Ñ	165	U	85	¿	168
6	54	E	69	o	111	ü	129	α	224
7	55	É	144	ó	162	ue		Γ	226
8	56	f	102	ò	149	Ü	154	π	227
9	57	F	70	ô	147	UE		Σ	228
a	97	g	103	O	79	v	118	σ	229
á	160	G	71	ö	148	V	86	µ	230
à	133	h	104	oe		w	119	τ	231
â	131	H	72	Ö	153	W	87	Φ	232
A	65	i	105	OE		x	120	Θ	233
ä	132	ï	139	p	112	X	88	Ω	234
ae		í	161	P	80	y	121	δ	235
Ä	142	ì	141	q	113	ÿ	152	∞	236
AE		î	140	Q	81	Y	89	φ	237
b	98	I	73	r	114	z	122	ε	238
B	66	j	106	R	82	Z	90	∩	239
c	99	J	74	s	115	å	134	ⁿ	252
ç	135	k	107	S	83	Å	143		
C	67	K	75	ß	225	æ	145		
Ç	128	l	108	ss		Æ	146		

DOS Code Page 437

Subtype #105 Collation Sequence - Nordan

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	E	69	l	108	ú	163	Å	143
0	48	É	144	M	77	ù	151	å	134
1	49	e	101	m	109	û	150	£	156
2	50	ë	137	N	78	V	86	*f*	159
3	51	é	130	Ñ	165	v	118	ª	166
4	52	è	138	n	110	W	87	º	167
5	53	ê	136	ñ	164	w	119	¿	168
6	54	F	70	O	79	X	88	α	224
7	55	f	102	o	111	x	120	ß	225
8	56	G	71	ó	162	Y	89	Γ	226
9	57	g	103	ò	149	Ü	154	π	227
A	65	H	72	ô	147	y	121	Σ	228
a	97	h	104	P	80	ÿ	152	σ	229
á	160	I	73	p	112	ü	129	µ	230
à	133	i	105	Q	81	Z	90	τ	231
â	131	ï	139	q	113	z	122	Φ	232
B	66	í	161	R	82	Æ	146	Θ	233
b	98	ì	141	r	114	Ä	142	Ω	234
C	67	î	140	S	83	æ	145	δ	235
Ç	128	J	74	s	115	ä	132	∞	236
c	99	j	106	T	84	Ø	157	φ	237
ç	135	K	75	t	116	Ö	153	ε	238
D	68	k	107	U	85	ø	155	∩	239
d	100	L	76	u	117	ö	148	n	252

DOS Code Page 865

Subtype #106 Collation Sequence - SwedFin

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	e	101	L	76	ù	151	ü	129
0	48	ë	137	m	109	û	150	Ü	154
1	49	é	130	M	77	U	85	¢	155
2	50	è	138	n	110	v	118	£	156
3	51	ê	136	ñ	164	V	86	¥	157
4	52	E	69	N	78	w	119	*f*	159
5	53	É	144	Ñ	165	W	87	ª	166
6	54	f	102	o	111	x	120	º	167
7	55	F	70	ó	162	X	88	¿	168
8	56	g	103	ò	149	y	121	α	224
9	57	G	71	ô	147	ÿ	152	ß	225
a	97	h	104	O	79	Y	89	Γ	226
á	160	H	72	p	112	z	122	π	227
à	133	i	105	P	80	Z	90	Σ	228
â	131	ï	139	q	113	æ	145	σ	229
A	65	í	161	Q	81	Æ	146	µ	230
b	98	ì	141	r	114	ϕ	237	τ	231
B	66	î	140	R	82	Φ	232	Θ	233
c	99	I	73	s	115	å	134	Ω	234
ç	135	j	106	S	83	Å	143	δ	235
C	67	J	74	t	116	ä	132	∞	236
Ç	128	k	107	T	84	Ä	142	ε	238
d	100	K	75	u	117	ö	148	∩	239
D	68	l	108	ú	163	Ö	153	ⁿ	252

DOS Code Page 437

Subtype #139 Collation Sequence - Denmark

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	AE		I	73	Ô	212	W	87
¡	161	b	98	í	237	õ	245	x	120
¿	191	B	66	Í	205	Õ	213	X	88
¤	164	c	99	ì	236	p	112	y	121
£	163	C	67	Ì	204	P	80	Y	89
¥	165	ç	231	î	238	q	113	ý	253
µ	181	Ç	199	Î	206	Q	81	Ý	221
0	48	d	100	ï	239	r	114	ÿ	255
1	49	D	68	Ï	207	R	82	ü	252
2	50	ð	240	j	106	s	115	Ü	220
3	51	Ð	208	J	74	S	83	z	122
4	52	e	101	k	107	ss		Z	90
5	53	E	69	K	75	ß	223	æ	230
6	54	é	233	l	108	t	116	Æ	198
7	55	É	201	L	76	T	84	ä	228
8	56	è	232	m	109	þ	222	Ä	196
9	57	È	200	M	77	Þ	254	ø	248
a	97	ê	234	n	110	u	117	Ø	216
A	65	Ê	202	N	78	U	85	ö	246
á	225	ë	235	ñ	241	ú	250	Ö	214
Á	193	Ë	203	Ñ	209	Ú	218	å	229
à	224	f	102	o	111	ù	249	Å	197
À	192	F	70	O	79	Ù	217	aa	
â	226	g	103	ó	243	û	251	Aa	
Â	194	G	71	Ó	211	Û	219	aA	
ã	227	h	104	ò	242	v	118	AA	
Ã	195	H	72	Ò	210	V	86		
ae		i	105	ô	244	w	119		

ISO8859.1 (Latin-1)

Subtype #140 Collation Sequence - Netherlands

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Å	197	ë	235	N	78	t	116
¡	161	ä	228	Ë	203	ñ	241	T	84
¿	191	Ä	196	f	102	Ñ	209	þ	254
¤	164	ã	227	F	70	o	111	Þ	222
¢	162	Ã	195	g	103	O	79	u	117
£	163	ae		G	71	ó	243	U	85
¥	165	AE		h	104	Ó	211	ú	250
µ	181	æ	230	H	72	ò	242	Ú	218
0	48	Æ	198	i	105	Ò	210	ù	249
1	49	b	98	I	73	ô	244	Ù	217
2	50	B	66	í	237	Ô	212	û	251
3	51	c	99	Í	205	õ	246	Û	219
4	52	C	67	ì	236	Õ	214	ũ	252
5	53	ç	231	Ì	204	ō	245	Ũ	220
6	54	Ç	199	î	238	Ō	213	v	118
7	55	d	100	Î	206	ø	248	V	86
8	56	D	68	ï	239	Ø	216	w	119
9	57	ð	240	Ï	207	p	112	W	87
a	97	Ð	208	j	106	P	80	x	120
A	65	e	101	J	74	q	113	X	88
á	225	E	69	k	107	Q	81	y	121
Á	193	é	233	K	75	r	114	Y	89
à	224	É	201	l	108	R	82	ý	253
À	192	è	232	L	76	s	115	Ý	221
â	226	È	200	m	109	S	83	ỹ	255
Â	194	ê	234	M	77	ss		z	122
å	229	Ê	202	n	110	ß	223	Z	90

Subtype #141 Collation Sequence - Finland

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Ã	195	h	104	Ó	211	û	251
¡	161	ae		H	72	ò	242	Û	219
¿	191	AE		i	105	Ò	210	v	118
¤	164	b	98	I	73	ô	244	V	86
¢	162	B	66	í	237	Ô	212	w	119
£	163	c	99	Í	205	õ	245	W	87
¥	165	C	67	ì	236	Õ	213	x	120
µ	181	ç	231	Ì	204	p	112	X	88
0	48	Ç	199	î	238	P	80	y	121
1	49	d	100	Î	206	q	113	Y	89
2	50	D	68	ï	239	Q	81	ý	253
3	51	ð	240	Ï	207	r	114	Ý	221
4	52	Ð	208	j	106	R	82	ÿ	255
5	53	e	101	J	74	s	115	ũ	252
6	54	E	69	k	107	S	83	Ũ	220
7	55	é	233	K	75	ß	223	z	122
8	56	É	201	l	108	ss		Z	90
9	57	è	232	L	76	t	116	å	229
a	97	È	200	m	109	T	84	Å	197
A	65	ê	234	M	77	þ	254	ã	228
á	225	Ê	202	n	110	Þ	222	Ã	196
Á	193	ë	235	N	78	u	117	æ	230
à	224	Ë	203	ñ	241	U	85	Æ	198
À	192	f	102	Ñ	209	ù	249	õ	246
â	226	F	70	o	111	Ù	217	Õ	214
Â	194	g	103	O	79	ú	250	ø	248
ã	227	G	71	ó	243	Ú	218	Ø	216

ISO8859.1 (Latin-1)

Subtype #142 Collation Sequence - France

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Å	196	ē	235	N	78	t	116
¡	161	ā	228	Ē	203	ñ	241	T	84
¿	191	Ā	196	f	102	Ñ	209	þ	254
¤	164	ā	227	F	70	o	111	Þ	222
¢	162	Ā	195	g	103	O	79	u	117
£	163	ae		G	71	ó	243	U	85
¥	165	AE		h	104	Ó	211	ú	250
µ	181	æ	230	H	72	ò	242	Ú	218
0	48	Æ	198	i	105	Ò	210	ù	249
1	49	b	98	I	73	ô	244	Ù	217
2	50	B	66	í	237	Ô	212	û	251
3	51	c	99	Í	205	õ	246	Û	219
4	52	C	67	ì	236	Õ	214	ū	252
5	53	ç	231	Ì	204	ō	245	Ū	220
6	54	Ç	199	î	238	Ő	213	v	118
7	55	d	100	Î	206	ø	248	V	86
8	56	D	68	ī	239	Ø	216	w	119
9	57	ð	240	Ī	207	p	112	W	87
a	97	Ð	208	j	106	P	80	x	120
A	65	e	101	J	74	q	113	X	88
á	225	E	69	k	107	Q	81	y	121
Á	193	é	233	K	75	r	114	Y	89
à	224	É	201	l	108	R	82	ý	253
À	192	è	232	L	76	s	115	Ý	221
â	226	È	200	m	109	S	83	ÿ	255
Â	194	ê	234	M	77	ss		z	122
å	229	Ê	202	n	110	ß	223	Z	90

ISO8859.1 (Latin-1)

Subtype #143 Collation Sequence - Canada

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Å	197	ë	235	N	78	t	116
¡	161	ã	228	Ë	203	ñ	241	T	84
¿	191	Ã	196	f	102	Ñ	209	þ	254
¤	164	ã	227	F	70	o	111	Þ	222
¢	162	Ã	195	g	103	O	79	u	117
£	163	ae		G	71	ó	243	U	85
¥	165	AE		h	104	Ó	211	ú	250
µ	181	æ	230	H	72	ò	242	Ú	218
0	48	Æ	198	i	105	Ò	210	ù	249
1	49	b	98	I	73	ô	244	Ù	217
2	50	B	66	í	237	Ô	212	û	251
3	51	c	99	Í	205	õ	246	Û	219
4	52	C	67	ì	236	Õ	214	ũ	252
5	53	ç	231	Ì	204	õ	245	Û	220
6	54	Ç	199	î	238	Õ	213	v	118
7	55	d	100	Î	206	ø	248	V	86
8	56	D	68	ï	239	Ø	216	w	119
9	57	đ	240	Ï	207	p	112	W	87
a	97	Đ	208	j	106	P	80	x	120
A	65	e	101	J	74	q	113	X	88
á	225	E	69	k	107	Q	81	y	121
Á	193	é	233	K	75	r	114	Y	89
à	224	É	201	l	108	R	82	ý	253
À	192	è	232	L	76	s	115	Ý	221
â	226	È	200	m	109	S	83	ÿ	255
Â	194	ê	234	M	77	ss		z	122
å	229	Ê	202	n	110	ß	223	Z	90

ISO8859.1 (Latin-1)

Subtype #144 Collation Sequence - Germany

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Å	197	ë	235	N	78	t	116
¡	161	ä	228	Ë	203	ñ	241	T	84
¿	191	Ä	196	f	102	Ñ	209	þ	254
¤	164	ã	227	F	70	o	111	Þ	222
¢	162	Ã	195	g	103	O	79	u	117
£	163	ae		G	71	ó	243	U	85
¥	165	AE		h	104	Ó	211	ù	249
µ	181	æ	230	H	72	ò	242	Ù	217
0	48	Æ	198	i	105	Ò	210	ú	250
1	49	b	98	I	73	ô	244	Ú	218
2	50	B	66	í	237	Ô	212	û	251
3	51	c	99	Í	205	õ	246	Û	219
4	52	C	67	ì	236	Õ	214	ũ	252
5	53	ç	231	Ì	204	õ	245	Ũ	220
6	54	Ç	199	î	238	Õ	213	v	118
7	55	d	100	Î	206	ø	248	V	86
8	56	D	68	ï	239	Ø	216	w	119
9	57	ð	240	Ï	207	p	112	W	87
a	97	Ð	208	j	106	P	80	x	120
A	65	e	101	J	74	q	113	X	88
á	225	E	69	k	107	Q	81	y	121
Á	193	é	233	K	75	r	114	Y	89
à	224	É	201	l	108	R	82	ý	253
À	192	è	232	L	76	s	115	Ý	221
â	226	È	200	m	109	S	83	ÿ	255
Â	194	ê	234	M	77	ss		z	122
å	229	Ê	202	n	110	ß	223	Z	90

ISO8859.1 (Latin-1)

Subtype #145 Collation Sequence - Iceland

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Ä	196	f	102	Ñ	209	û	251
¡	161	ã	227	F	70	o	111	Û	219
¿	191	Ã	195	g	103	O	79	ũ	252
¤	164	ae		G	71	ò	242	Ũ	220
¢	162	AE		h	104	Ò	210	ú	250
£	163	á	225	H	72	ô	244	Ú	218
¥	165	Á	193	i	105	Ô	212	v	118
µ	181	b	98	I	73	õ	245	V	86
0	48	B	66	ì	236	Õ	213	w	119
1	49	c	99	Ì	204	ó	243	W	87
2	50	C	67	î	238	Ó	211	x	120
3	51	ç	231	Î	206	p	112	X	88
4	52	Ç	199	ï	239	P	80	y	121
5	53	d	100	Ï	207	q	113	Y	89
6	54	D	68	í	237	Q	81	ÿ	255
7	55	ð	240	Í	205	r	114	ý	253
8	56	Ð	208	j	106	R	82	Ý	221
9	57	e	101	J	74	s	115	z	122
a	97	E	69	k	107	S	83	Z	90
A	65	è	232	K	75	ss		þ	254
à	224	È	200	l	108	ß	223	Þ	222
À	192	ê	234	L	76	t	116	æ	230
â	226	Ê	202	m	109	T	84	Æ	198
Â	194	ë	235	M	77	u	117	ö	246
å	229	Ë	203	n	110	U	85	Ö	214
Å	197	é	233	N	78	ù	249	ø	248
ã	228	É	201	ñ	241	Ù	217	Ø	216

ISO8859.1 (Latin-1)

Subtype #146 Collation Sequence - Italy

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Å	197	ë	235	N	78	t	116
¡	161	ä	228	Ë	203	ñ	241	T	84
¿	191	Ä	196	f	102	Ñ	209	þ	254
¤	164	ã	227	F	70	o	111	Þ	222
¢	162	Ã	195	g	103	O	79	u	117
£	163	ae		G	71	ó	243	U	85
¥	165	AE		h	104	Ó	211	ú	250
µ	181	æ	230	H	72	ò	242	Ú	218
0	48	Æ	198	i	105	Ò	210	ù	249
1	49	b	98	I	73	ô	244	Ù	217
2	50	B	66	í	237	Ô	212	û	251
3	51	c	99	Í	205	õ	246	Û	219
4	52	C	67	ì	236	Õ	214	ü	252
5	53	ç	231	Ì	204	ō	245	Ü	220
6	54	Ç	199	î	238	Ō	213	v	118
7	55	d	100	Î	206	ø	248	V	86
8	56	D	68	ï	239	Ø	216	w	119
9	57	ð	240	Ï	207	p	112	W	87
a	97	Ð	208	j	106	P	80	x	120
A	65	e	101	J	74	q	113	X	88
á	225	E	69	k	107	Q	81	y	121
Á	193	é	233	K	75	r	114	Y	89
à	224	É	201	l	108	R	82	ý	253
À	192	è	232	L	76	s	115	Ý	221
â	226	È	200	m	109	S	83	ÿ	255
Â	194	ê	234	M	77	ss		z	122
å	229	Ê	202	n	110	ß	223	Z	90

ISO8859.1 (Latin-1)

Subtype #148 Collation Sequence - Norway

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Ã	195	H	72	Ò	210	Û	219
¡	161	ae		i	105	ð	244	ũ	252
¿	191	AE		I	73	Ô	212	Ũ	220
¤	164	b	98	í	237	õ	246	v	118
£	163	B	66	Í	205	Õ	214	V	86
¥	165	c	99	ì	236	ō	245	w	119
µ	181	C	67	Ì	204	Ō	213	W	87
0	48	ç	231	î	238	p	112	x	120
1	49	Ç	199	Î	206	P	80	X	88
2	50	d	100	ï	239	q	113	y	121
3	51	D	68	Ï	207	Q	81	Y	89
4	52	đ	240	j	106	r	114	ý	253
5	53	Đ	208	J	74	R	82	Ý	221
6	54	e	101	k	107	s	115	ỹ	255
7	55	E	69	K	75	S	83	z	122
8	56	é	233	l	108	ss		Z	90
9	57	É	201	L	76	ß	223	æ	230
a	97	è	232	m	109	t	116	Æ	198
A	65	È	200	M	77	T	84	ø	248
á	225	ê	234	n	110	þ	254	Ø	216
Á	193	Ê	202	N	78	Þ	222	å	229
à	224	ë	235	ñ	241	u	117	Å	197
À	192	Ë	203	Ñ	209	U	85	aa	
â	226	f	102	o	111	ú	250	Aa	
Â	194	F	70	O	79	Ú	218	aA	
ã	228	g	103	ó	243	ù	249	AA	
Ã	196	G	71	Ó	211	Ù	217		
ã	227	h	104	ò	242	û	251		

ISO8859.1 (Latin-1)

Subtype #149 Collation Sequence - Spain

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Ã	196	Ẽ	203	N	78	þ	254
¡	161	ã	227	f	102	ñ	241	Þ	222
¿	191	Ã	195	F	70	Ñ	209	u	117
¤	164	ae		g	103	o	111	U	85
¢	162	AE		G	71	O	79	ú	250
£	163	æ	230	h	104	ó	243	Ú	218
¥	165	Æ	198	H	72	Ó	211	ù	249
µ	181	b	98	i	105	ò	242	Ù	217
0	48	B	66	I	73	Ò	210	û	251
1	49	c	99	í	237	ô	244	Û	219
2	50	C	67	Í	205	Ô	212	ũ	252
3	51	ç	231	ì	236	õ	246	Ũ	220
4	52	Ç	199	Ì	204	Õ	214	v	118
5	53	ch		î	238	ō	245	V	86
6	54	Ch		Î	206	Ō	213	w	119
7	55	CH		ï	239	ø	248	W	87
8	56	d	100	Ï	207	Ø	216	x	120
9	57	D	68	j	106	p	112	X	88
a	97	ð	240	J	74	P	80	y	121
A	65	Ð	208	k	107	q	113	Y	89
á	225	e	101	K	75	Q	81	ý	253
Á	193	E	69	l	108	r	114	Ý	221
à	224	é	233	L	76	R	82	ÿ	255
À	192	É	201	ll		s	115	z	122
â	226	è	232	Ll		S	83	Z	90
Â	194	È	200	LL		ss			
å	229	ê	234	m	109	ß	223		
Å	197	Ê	202	M	77	t	116		
ã	228	ë	235	n	110	T	84		

ISO8859.1 (Latin-1)

Subtype #151 Collation Sequence - Sweden

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Ã	195	h	104	Ó	211	û	251
¡	161	ae		H	72	ò	242	Û	219
¿	191	AE		i	105	Ò	210	v	118
¤	164	b	98	I	73	ô	244	V	86
¢	162	B	66	í	237	Ô	212	w	119
£	163	c	99	Í	205	õ	245	W	87
¥	165	C	67	ì	236	Õ	213	x	120
µ	181	ç	231	Ì	204	p	112	X	88
0	48	Ç	199	î	238	P	80	y	121
1	49	d	100	Î	206	q	113	Y	89
2	50	D	68	ï	239	Q	81	ý	253
3	51	ð	240	Ï	207	r	114	Ý	221
4	52	Ð	208	j	106	R	82	ÿ	255
5	53	e	101	J	74	s	115	ũ	252
6	54	E	69	k	107	S	83	Ũ	220
7	55	è	232	K	75	ss		z	122
8	56	È	200	l	108	ß	223	Z	90
9	57	é	233	L	76	t	116	å	229
a	97	É	201	m	109	T	84	Å	197
A	65	ê	234	M	77	þ	254	ã	228
á	225	Ê	202	n	110	Þ	222	Ã	196
Á	193	ë	235	N	78	u	117	æ	230
à	224	Ë	203	ñ	241	U	85	Æ	198
À	192	f	102	Ñ	209	ú	250	õ	246
â	226	F	70	o	111	Ú	218	Õ	214
Â	194	g	103	O	79	ù	249	ø	248
ã	227	G	71	ó	243	Ù	217	Ø	216

ISO8859.1 (Latin-1)

Subtype #152 Collation Sequence - UK

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Å	197	ē	235	N	78	t	116
¡	161	ā	228	Ē	203	ñ	241	T	84
¿	191	Ā	196	f	102	Ñ	209	þ	254
¤	164	ă	227	F	70	o	111	Þ	222
¢	162	Ă	195	g	103	O	79	u	117
£	163	ae		G	71	ó	243	U	85
¥	165	AE		h	104	Ó	211	ú	250
µ	181	æ	230	H	72	ò	242	Ú	218
0	48	Æ	198	i	105	Ò	210	ù	249
1	49	b	98	I	73	ô	244	Ù	217
2	50	B	66	í	237	Ô	212	û	251
3	51	c	99	Í	205	ō	246	Û	219
4	52	C	67	ì	236	Ō	214	ū	252
5	53	ç	231	Ì	204	ő	245	Ū	220
6	54	Ç	199	î	238	Ő	213	v	118
7	55	d	100	Î	206	ø	248	V	86
8	56	D	68	ī	239	Ø	216	w	119
9	57	đ	240	Ī	207	p	112	W	87
a	97	Đ	208	j	106	P	80	x	120
A	65	e	101	J	74	q	113	X	88
á	225	E	69	k	107	Q	81	y	121
Á	193	é	233	K	75	r	114	Y	89
à	224	É	201	l	108	R	82	ý	253
À	192	è	232	L	76	s	115	Ý	221
â	226	È	200	m	109	S	83	ÿ	255
Â	194	ê	234	M	77	ss		z	122
å	229	Ê	202	n	110	ß	223	Z	90

ISO8859.1 (Latin-1)

Subtype #153 Collation Sequence - US

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	i	105	V	86	Ë	203	ç	231
0	48	I	73	w	119	Ì	204	è	232
1	49	j	106	W	87	Í	205	é	233
2	50	J	74	x	120	Î	206	ê	234
3	51	k	107	X	88	Ï	207	ë	235
4	52	K	75	y	121	Ð	208	ì	236
5	53	l	108	Y	89	Ñ	209	í	237
6	54	L	76	z	122	Ò	210	î	238
7	55	m	109	Z	90	Ó	211	ï	239
8	56	M	77	¡	161	Ô	212	ð	240
9	57	n	110	¢	162	Õ	213	ñ	241
a	97	N	78	£	163	Ö	214	ò	242
A	65	o	111	¤	164	Ø	216	ó	243
b	98	O	79	¥	165	Ù	217	ô	244
B	66	p	112	µ	181	Ú	218	õ	245
c	99	P	80	¿	191	Û	219	ö	246
C	67	q	113	À	192	Ü	220	ø	248
d	100	Q	81	Á	193	Ý	221	ù	249
D	68	r	114	Â	194	Þ	222	ú	250
e	101	R	82	Ã	195	ß	223	û	251
E	69	s	115	Ä	196	à	224	ü	252
f	102	S	83	Å	197	á	225	ý	253
F	70	t	116	Æ	198	â	226	þ	254
g	103	T	84	Ç	199	ã	227	ÿ	255
G	71	u	117	È	200	ä	228		
h	104	U	85	É	201	å	229		
H	72	v	118	Ê	202	æ	230		

ISO8859.1 (Latin-1)

Subtype #154 Collation Sequence - Portugal

CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE	CHAR	CODE
	32	Å	197	ë	235	N	78	t	116
¡	161	ã	228	Ë	203	ñ	241	T	84
¿	191	Ã	196	f	102	Ñ	209	þ	254
¤	164	ā	227	F	70	o	111	Þ	222
¢	162	Ā	195	g	103	O	79	u	117
£	163	ae		G	71	ó	243	U	85
¥	165	AE		h	104	Ó	211	ú	250
µ	181	æ	230	H	72	ò	242	Ú	218
0	48	Æ	198	i	105	Ò	210	ù	249
1	49	b	98	I	73	ô	244	Ù	217
2	50	B	66	í	237	Ô	212	û	251
3	51	c	99	Í	205	õ	246	Û	219
4	52	C	67	ì	236	Õ	214	ũ	252
5	53	ç	231	Ì	204	ő	245	Ũ	220
6	54	Ç	199	î	238	Ő	213	v	118
7	55	d	100	Î	206	ø	248	V	86
8	56	D	68	ī	239	Ø	216	w	119
9	57	đ	240	Ī	207	p	112	W	87
a	97	Đ	208	j	106	P	80	x	120
A	65	e	101	J	74	q	113	X	88
á	225	E	69	k	107	Q	81	y	121
Á	193	é	233	K	75	r	114	Y	89
à	224	É	201	l	108	R	82	ý	253
À	192	è	232	L	76	s	115	Ý	221
â	226	È	200	m	109	S	83	ÿ	255
Â	194	ê	234	M	77	ss		z	122
å	229	Ê	202	n	110	ß	223	Z	90

ISO8859.1 (Latin-1)

Appendix D

V3.3 Platform, OS, and Language Support

Platform	O/S	Multi-Client Serv.	Multi-Thread Serv.	gpre Language Support	Networks
Apollo DN3/4xxx, HP 9000/400	SR10.3	Y	Y	C++, C, Pas, FOR, ADA	TCP/IP, MBX
Apollo DN10000	SR10.2-3	Y	Y	C, Pas, FOR, ADA	TCP/IP, MBX
Data General AViiON	DG-UX 5.4			C++, C, COB, FOR	TCP/IP
HP 9000/300,400	HP-UX 8.0			C++, C, FOR, ADA	TCP/IP
HP 9000/700	HP-UX 8.0.5			C, COB, FOR	TCP/IP
HP 9000/600, 800	HP-UX 8.0.5			C, FORTRAN	TCP/IP
IBM 386/486 Compatibles	SCO Unix 3.2.4			C	TCP/IP
IBM RS/6000	AIX 3.2			C, COB, FOR	TCP/IP
IMP, XTM	UNIPLUS+ V3.1			C	TCP/IP
Motorola Delta	Sys V/68 Rel 3			C	TCP/IP
Silicon Graphics	IRIX 3.3.1, 4.0.1			C, FOR	TCP/IP
Sun-4/SPARC	SunOS 4.1.2-3	Y		C++, C, FOR, ADA, COB	TCP/IP
DEC MIPS Ultrix	Ultrix 4.2			C	TCP/IP DECnet
DEC VAX/VMS	VAX/VMS V5.3-5			C, Pas, FOR,COB, ADA	TCP/IP, DECnet

Index

T

U

V

W

X