Utility Programs

Sybase® Adaptive Server® Enterprise

Version 12.0

UNIX Platforms

Document ID: 36124-01-1200-01

Last revised: October 1999

Sybase, Inc., 6475 Christie Avenue, Emeryville, CA 94608.

Contents

About This Book ..... **ix**

CHAPTER 1 **Using the isql Utility** **1**
- Starting and Stopping the isql Utility 2
- How to Use Transact-SQL in the isql Utility 3
  - Formatting isql Output 3
  - Correcting Input 4
  - set Options That Affect Output 5
- Changing the Command Terminator 6
- Performance Statistics Interaction with Command Terminator Values 7
- Setting the Network Packet Size 8
  - Setting the Network Packet Size for the Client 8
- Input and Output Files 9
  - UNIX Command Line Redirection 9

CHAPTER 2 **Using bcp to Transfer Data to and from Adaptive Server** **11**
- Methods for Moving Data 12
  - Using bcp to Import and Export Data 12
- Requirements for Using bcp 14
  - Basic Information Required 14
  - Permissions Needed to Copy Data 14
  - Pre-Transfer Task 14
- bcp Modes 16
- bcp Performance Issues 17
  - Using Fast bcp or Slow bcp 17
  - Steps for Copying In Data with Fast bcp 19
  - Bulk Copying Data into Partitioned Tables 19
  - Using Parallel Bulk Copy to Copy Data Into a Specific Partition 22
- Using the bcp Options 29
  - Using the Default Formats 29
  - Changing Terminators From the Command Line 30
- Changing the Defaults: Interactive bcp 31

File Storage Type 31
Prefix Length 33
Field Length 34
Field and Row Terminators 35
Using Format Files 37
Elements of the bcp Format File 37
Examples: Copying Out Data Interactively 41
Copying Out Data with Field Lengths 41
Copying Out Data with Delimiters 42
Examples: Copying In Data Interactively 44
Copying In Data with Field Lengths 44
Copying In Data with Delimiters 46
Copying In Data with a Format File 46
Using bcp with Alternate Languages 48
Copy In and Batch Files 49
Improving Recoverability 49
Batches and Partitioned Tables 50
Copy Out and text and image Data 51
Specifying a Network Packet Size 52
Copy In and Error Files 53
Copy Out and Error Files 54
Data Integrity: Defaults, Rules, and Triggers 55
Defaults and Datatypes 55
Rules and Triggers 55
How bcp Differs from Other Utilities 56

CHAPTER 3 Using dsedit 57
Getting Started with dsedit 58
Starting dsedit 58
Opening an Editing Session 59
Adding, Viewing, and Editing Server Entries 61
Adding or Editing Network Transport Addresses 62
SPX/IPX Addresses 62
Troubleshooting dsedit 64
The dsedit Utility Does Not Start 64
Error Message: “Unable to open X display” 64
Cannot Add, Modify, or Delete Server Entries 65

CHAPTER 4 Using dscp 67
Getting Started with dscp 68
Starting dscp 68
Getting Help with dscp 68
Using a dscp Session 68

Working with Server Entries 70
Adding and Modifying Server Entries 70
Copying Server Entries 73
Listing Server Entries 74
Viewing Contents of Server Entries 74
Deleting Server Entries 75
Exiting dscp 76
Quick Reference for dscp Utility Commands 77

CHAPTER 5 Utility Commands Reference 79
Starting A Utility Program 80
Utilities Quick Reference 82
Installation or Configuration Utilities 82
Utilities for Languages, Character Sets and Sort Orders 83
Utilities to Start Servers 83
Database Creation and Manipulation Utilities 83
Utilities to Gather Information 84
backupserver 85
bcp 90
bcp_dce 105
buildmaster 122
charset 125
dataserver 126
defncopy 129
defncopy_dce 134
dscp 139
dscp_dce 140
dsedit 141
dsedit_dce 142
extractjava 143
installjava 146
isql 150
isql_dce 158
langinstall 166
optdiag 168
showserver 175
sqlloc 176
sqllocres 177
sqlupgrade 178
sqlupgraderes 179
srvbuild 180
srvbuildres 181
startserver 182
sybload 185

sybsetup 187

About This Book

Utility Programs for UNIX Platforms is a guide to the Sybase® Adaptive Server® Enterprise utility programs available for UNIX platforms. Utility programs are commands that you invoke directly from the operating system.

Audience

This manual is for anyone using Transact-SQL® and Adaptive Server Enterprise version 12. It assumes that you have the basic knowledge to use the UNIX platforms operating systems and Adaptive Server.

How to Use This Book

This manual includes the following chapters:

Chapter 1,

- "Using the isql Utility" – discusses how to use the interactive SQL (isql) utility that allows access to SQL from your operating system.
- Chapter 2, "Using bcp to Transfer Data to and from Adaptive Server"– discusses, in detail, the Bulk Copy (**bcp**) utility which you use to move data between Adaptive Server and an operating system file.
- Chapter 3, "Using dsedit" – explains how to use the Directory Services Editor (**dsedit)** utility in X-Windows to view and edit server entries in the interfaces file.
- Chapter 4, "Using dscp" – explains how to use the **dscp** utility to view and edit server entries in the interfaces file.
- Chapter 5, "Utility Commands Reference" – lists and describes the utility commands that you use to manage and maintain your databases and Adaptive Server Enterprise.

The examples in this manual are based on the *pubs2* sample database. Ask your System Administrator how to access a clean copy of *pubs2*.

Related documents

The following documents comprise the Sybase Adaptive Server Enterprise documentation:

- The release bulletin for your platform – contains last-minute information that was too late to be included in the books.

A more recent version of this release bulletin may be available on the World Wide Web. To check for critical product or document information added after the release of the product CD, use the Sybase Technical Library Product Manuals web site.

To access release bulletins at the Technical Library Product Manuals web site:

a Go to Product Manuals at http://sybooks.sybase.com.

b Select a product family link.

c Select a product link.

d From the Collection list in the left frame, select the "platform-specific" link for the product and version you are interested in.

e From the list of individual documents in the right frame, select the link to the release bulletin for your platform.

Browse the document online or download a PDF version by clicking the PDF button at the bottom of the left frame.

- The Adaptive Server installation guide for your platform – describes installation, upgrade, and configuration procedures for all Adaptive Server and related Sybase products.
- *What's New in Adaptive Server Enterprise?* – describes the new features in Adaptive Server version 12, the system changes added to support those features, and the changes that may affect your existing applications.
- *Transact-SQL User's Guide* – documents Transact-SQL, Sybase's enhanced version of the relational database language. This manual serves as a textbook for beginning users of the database management system. This manual also contains descriptions of the *pubs2* and *pubs3* sample databases.
- *System Administration Guide* – provides in-depth information about administering servers and databases. This manual includes instructions and guidelines for managing physical resources, security, user and system databases, and specifying character conversion, international language, and sort order settings.
- *Adaptive Server Reference Manual* – contains detailed information about all Transact-SQL commands, functions, procedures, and datatypes. This manual also contains a list of the Transact-SQL reserved words and definitions of system tables.

- *Performance and Tuning Guide* – explains how to tune Adaptive Server for maximum performance. This manual includes information about database design issues that affect performance, query optimization, how to tune Adaptive Server for very large databases, disk and cache issues, and the effects of locking and cursors on performance.
- The Utility Programs manual for your platform – documents the Adaptive Server utility programs, such as isql and bcp, which are executed at the operating system level.
- *Error Messages and Troubleshooting Guide* – explains how to resolve frequently occurring error messages and describes solutions to system problems frequently encountered by users.
- *Component Integration Services User's Guide* – explains how to use the Adaptive Server Component Integration Services feature to connect remote Sybase and non-Sybase databases.
- *Java in Adaptive Server Enterprise* – describes how to install and use Java classes as datatypes and user-defined functions in the Adaptive Server database.
- *Using Sybase Failover in a High Availability System* – provides instructions for using Sybase's Failover to configure an Adaptive Server as a companion server in a high availability system.
- *Using Adaptive Server Distributed Transaction Management Features* – explains how to configure, use, and troubleshoot Adaptive Server DTM features in distributed transaction processing environments.
- *XA Interface Integration Guide for CICS, Encina, and TUXEDO* – provides instructions for using Sybase's DTM XA Interface with X/Open XA transaction managers.
- *Adaptive Server Glossary* – defines technical terms used in the Adaptive Server documentation.

Other sources of information

Use the Sybase Technical Library CD and the Technical Library Product Manuals web site to learn more about your product:

- Technical Library CD contains product manuals and technical documents and is included with your software. The DynaText browser (included on the Technical Library CD) allows you to access technical information about your product in an easy-to-use format.

  Refer to the *Technical Library Installation Guide* in your documentation package for instructions on installing and starting Technical Library.

- Technical Library Product Manuals web site is an HTML version of the Technical Library CD that you can access using a standard web browser. In addition to product manuals, you'll find links to the Technical Documents web site (formerly known as Tech Info Library), the Solved Cases page, and Sybase/Powersoft newsgroups.

  To access the Technical Library Product Manuals web site, go to Product Manuals at http://sybooks.sybase.com.

Sybase certifications on the web

Technical documentation at the Sybase web site is updated frequently.

❖ **For the latest information on product certifications and/or the EBF Rollups:**

1. Point your web browser to Technical Documents at http://techinfo.sybase.com.
2. In the Browse section, click on What's Hot.
3. Select links to Certification Reports and EBF Rollups, as well as links to Technical Newsletters, online manuals, and so on.

❖ **If you are a registered SupportPlus user:**

1. Point your web browser to Technical Documents at http://techinfo.sybase.com.
2. In the Browse section, click on What's Hot.
3. Click on EBF Rollups.

   You can research EBFs using Technical Documents, and you can download EBFs using Electronic Software Distribution (ESD).
4. Follow the instructions associated with the SupportPlus^{SM} Online Services entries.

❖ **If you are not a registered SupportPlus user, and you want to become one:**

You can register by following the instructions on the Web.

To use SupportPlus, you need:

1. A Web browser that supports the Secure Sockets Layer (SSL), such as Netscape Navigator 1.2 or later
2. An active support license
3. A named technical support contact
4. Your user ID and password

❖ **Whether or not you are a registered SupportPlus user:**

You may use Sybase's Technical Documents. Certification Reports are among the features documented at this site.

1 Point your web browser to Technical Documents at http://techinfo.sybase.com

2 In the Browse section, click on What's Hot.

3 Click on the topic that interests you.

Conventions

In the regular text of this document, the names of files and directories appear in italics, for example:

$SYBASE

Table 1 details the typographic (font and syntax) conventions as used in this document.

Table 1: Font and syntax conventions for this document

Element	Example
Command names, command option names, utility names, utility flags, and other keywords are bold.	dsedit
Database names, datatypes, file names and path names are in *italics*.	*master* database *$SYBASE/bin* **Note** Be sure to substitute your Sybase installation drive and directory for *$SYBASE*.
Variables, or words that stand for values that you fill in, are in *italics*.	`select` *column_name* `from` *table_name* `where` *search_conditions*
Parentheses must be typed as part of the command.	`compute row_aggregate (column_name)`
Curly braces indicate that at least one of the enclosed options is required by the command (see comma). **Note** Do not type the curly braces.	`{cheese, sauce}`

Element	Example
Brackets mean that choosing one or more of the enclosed options is optional. **Note** Do not type the brackets.	`[anchovies, pineapple, bell_peppers]`
The *vertical bar* means you may select only one of the options shown.	`{cash | check | credit}`
The *comma* means you may choose as many of the options shown as you like; be sure to separate multiple choices in a command with commas.	`[extra_cheese, avocados, sour_cream]`
An *ellipsis* (...) means that you can *repeat* the unit that the ellipsis follows as many times as you like.	`buy` *thing* `= price [cash | check | credit] [,` *thing* `= price [cash | check | credit] ]...` – You must buy at least one *thing* (item) and give its price. – You may choose a method of payment: one of the options enclosed in square brackets. – You may choose also to buy additional items: as many of them as you like. For each item you buy, provide its name, its price, and (optionally) a method of payment.

Syntax statements, which display the utility's syntax including all its options, appear as shown here:
 	
Or, for a utility with more options:
 	
Note In syntax statements, utilities and their syntax display in lowercase normal font, although the various parts of the syntax can be either normal font for flags (-c) and options (cash) or italics for user-supplied values (*physicalname*).
	`dscp [-p | -v]`
	
	`buildmaster [-d physicalname] [-c cntrltype] [-s size] [-m]`

Examples that illustrate utility commands appear in bold, as shown:	bcp -v
Examples that illustrate computer output appear as shown:	[See example that follows this table.

```
pub_id  pub_name                city        state
```

```
------- --------------------      ----------- -----
0736  New Age Books           Boston       MA
0877  Binnet & Hardley        Washington   DC
1389  Algodata Infosystems    Berkeley     CA

(3 rows affected)
```

If You Need Help

Each Sybase installation that has purchased a support contract has one or more designated people who are authorized to contact Sybase Technical Support. If you cannot resolve a problem using the manuals or online help, please have the designated person contact Sybase Technical Support or the Sybase subsidiary in your area.

CHAPTER 1

Using the *isql* Utility

This chapter describes the interactive SQL utility, **isql**.

Topics covered are:

Name	Page
Starting and Stopping the isql Utility	2
How to Use Transact-SQL in the isql Utility	3
Changing the Command Terminator	6
Performance Statistics Interaction with Command Terminator Values	7
Setting the Network Packet Size	8
Input and Output Files	9

Note To use Transact-SQL directly from the operating system with the **isql** utility program, you must have an account, or login, on Adaptive Server.

For a detailed description of **isql** syntax, see **installjava on page 146**.

Starting and Stopping the *isql* Utility

To start **isql**:

1 Enter the following command at the operating system prompt:

```
isql
```

The following prompt appears:

```
Password:
```

2 Enter your password.

The password does not appear on the screen as you type. The **isql** prompt appears:

```
1>
```

You can now issue Transact-SQL commands.

To exit **isql**:

Enter either of the following commands on a line by itself:

```
quit
exit
```

How to Use Transact-SQL in the *isql* Utility

isql sends Transact-SQL commands to Adaptive Server, formatting the results and printing them to standard output. There is no maximum size for an **isql** statement. For more information about using Transact-SQL, see the Transact-SQL User's Guide.

To terminate aTransact-SQL command, type the default command terminator "go" on a new line.

For example:

```
isql
Password:

1> use pubs2
2> go
1> select *
2> from authors
3> where city = "Oakland"
4> go
```

Formatting *isql* Output

Table 1-1 describes the options that change the format of **isql** output:

Table 1-1: Format options for isql

Option	Default	Meaning
-h *headers*	1	Number of rows to print between column headings
-s *colseparator*	Single space	Changes the column separator character
-w *columnwidth*	80 characters	Changes the line width

To include each command issued to **isql** in the output, use the **-e** option. To remove numbering and prompt symbols, use the **-n** option.

For example:

```
use pubs2
go
isql -e -n -o output
Password:

select *
from authors
where city = "Oakland"
go
```

```
 quit

cat output

select *
from authors
where city = "Oakland"
au_id     au_lname               au_fname
phone        address
city          state country    postalcode
----------- ---------------------------------------- -
-----------
--------
------------ ----------------------------------------
-------------------- ----- ------------ ----------
213-46-8915 Green                    Marjorie
          415 986-7020 309 63rd St. #411
          Oakland   CA     USA    94618
274-80-9391 Straight                 Dick
          415 834-2919 5420 College Av.
          Oakland   CA     USA    94609
724-08-9931 Stringer                 Dirk
          415 843-2991 5420 Telegraph Av.
          Oakland   CA     USA    94609
724-80-9391 MacFeather               Stearns
          415 354-7128 44 Upland Hts.
          Oakland   CA     USA    94612
756-30-7391 Karsen                   Livia
          415 534-9219 5720 McAuley St.
          Oakland   CA     USA    94609
```

Note that the *output* file does not include the command terminator.

Correcting Input

If you make an error when typing a Transact-SQL command, you can do one of the following:

- Press Ctrl-c or type the word “reset” on a line by itself.

  This clears the query buffer and returns the **isql** prompt.

- Type the name of your text editor on a line by itself.

  This opens a text file where you can edit the query. When you write and save the file, you are returned to **isql** and the corrected query appears. Type “go” to execute it.

set Options That Affect Output

Table 1-2 lists the **set** options that affect Transact-SQL output. For more information, see **set** in the *Adaptive Server Reference Manual*.

Table 1-2: set options that affect Transact-SQL output

set Option	Default	Meaning
char_convert	Off	Turns character set conversion off and on between Adaptive Server and a client; also starts a conversion between the server character set and a different client character set.
fipsflagger	Off	Warns when any Transact-SQL extensions to entry level SQL92 are used.This option does not disable the SQL extensions. Processing completes when you issue the non-ANSI SQL command.
flushmessage	Off	Sends messages as they are generated.
language	us_english	Sets the language for system messages.
nocount	Off	Turns off report of number of rows affected.
noexec	Off	Compiles each query but does not execute it; often used with showplan.
parseonly	Off	Checks the syntax of queries and returns error messages without compiling or executing the queries.
showplan	Off	Generates a description of the processing plan for a query; does not print results when used inside a stored procedure or trigger.
statistics io statistics time	Off	Displays performance statistics after each execution.
statistics subquerycache	Off	Displays the number of cache hits, misses, and rows in the subquery cache for each subquery.
textsize	32K	Controls the number of bytes of text or image data returned.

Changing the Command Terminator

If you include the command terminator argument (**-c**), you can choose your own terminator symbol; **go** is the default value for this option. Always enter the command terminator without blanks or tabs in front of it.

For example, to use a period as the command terminator, invoke **isql** as follows:

```
isql -c.
```

A sample **isql** session with this command terminator looks like this:

```
1> select name from sysusers
2> .
name
-----------
sandy
kim
leslie
(3 rows affected)
```

Using the **isql** command terminator option with scripts requires advance planning:

- Adaptive Server-supplied scripts, such as **installmaster**, use “go”. Do not change the command terminator for any session that uses these scripts.
- Your own scripts may already have “go” in them. Remember to update your scripts to include the terminator you plan to use.

Performance Statistics Interaction with Command Terminator Values

isql provides a performance statistics option (**-p**).

For example:

```
isql -p
1> select * from sysobjects
2> go
```

returns the following statistics:

```
Execution Time (ms.): 1000   Clock Time (ms.): 1000
1 xact:
```

This means that a single transaction took 100 ms. The clock time value reflects the entire transaction, which starts when Client-Library™ builds the query and ends when Client-Library returns the information from Adaptive Server.

You can gather performance statistics based on the execution of one or more transactions. To gather statistics on more than one transaction, specify a number after the command terminator. For example, the following command:

```
isql -p

1> select * from sysobjects
2> go 3
```

instructs Adaptive Server to execute three **select *** transactions and report the performance statistics. Adaptive Server returns:

```
Execution Time (ms.): 1000   Clock Time (ms.): 1000
Execution Time (ms.): 1000   Clock Time (ms.): 2000
Execution Time (ms.): 1000   Clock Time (ms.): 1000

Execution Time (ms.): 1000   Clock Time (ms.): 4000
3xact:
```

Setting the Network Packet Size

Setting the correct network packet size can greatly increase the performance of Adaptive Server.

Setting the Network Packet Size for the Client

The **-A size** option specifies the network packet size to use for an **isql** session. For example:

```
isql -A 2048
```

sets the packet size to 2048 bytes for this **isql** session. To check your network packet size, type:

```
select * from sysprocesses
```

The value for this **isql** session appears under the *network_pktsz* heading in the *sysprocesses* table.

See the *System Administration Guide* for more information about setting the network packet size.

Input and Output Files

You can specify input and output files on the command line with the **-i** and **-o** options.

isql does not provide formatting options for the output. However, you can use the **-n** option to eliminate the **isql** prompts and other tools to reformat the output.

If you use the **-e** option, **isql** echoes the input to output. The resulting output file contains both the queries and their results.

UNIX Command Line Redirection

The UNIX redirection symbols, " <" and ">", provide a similar mechanism to the **-i** and **-o** options, as follows:

```
isql -Usa < input > output
```

You can direct **isql** to take input from the terminal, as shown in the following example:

```
isql -Usa -Ppassword -Sserver_name << EOF > output
use pubs2
go
select * from table
go
EOF
```

"<<EOF" instructs **isql** to take input from the terminal up to the string "EOF." You can replace "EOF" with any character string. Similarly, the following example signals the end of input with Ctrl-d:

```
isql -Usa << > output
```

CHAPTER 2

Using *bcp* to Transfer Data to and from Adaptive Server

This chapter explains how to use the bulk copy utility, **bcp**, to move data between Adaptive Server and an operating system file. Topics covered are:

For a detailed description of the **bcp** systax, see **bcp on page 90**.

Methods for Moving Data

You can use the following methods to move data to and from your Adaptive Server databases:

- Use **bcp** as a standalone program from the operating system. This chapter provides instructions for this method.
- Use Client-Library, which calls bulk library routines.

  For more information about the Client-Library, see the *Open Client and Open Server Common Libraries Reference Manual*.

Using *bcp* to Import and Export Data

Because Transact-SQL commands cannot transfer data in bulk, you must use **bcp** for any large transfers. You can use **bcp** to:

- Import data that was previously associated with another program, such as the records from another database management system. This is the most common use for **bcp**.

  Before using **bcp**, you must create a file of the records you want to import. The general steps are:

  a Put the data to transfer into an operating system file.

  b Run **bcp** from the operating system command line.
- Move tables between Adaptive Servers or between Adaptive Server and other data sources that can produce an operating system file.
- Copy out data from a view. For a description of the syntax for using **bcp** to copy out from a view, see **bcp on page 90**.
- Copy out data from a view. For a description of the syntax for using **bcp_dce** to copy out from a view, see **bcp_dce on page 105**.

  Note You cannot use **bcp** to copy *in* data to a view.

- Transfer data for use with other programs, for example, with a spreadsheet program. The general steps to transfer data are:

  a Use **bcp** to move the data from Adaptive Server into an operating system file from which the other program imports the data.

b When you finish using your data with the other program, copy it into an operating system file, and then use **bcp** to copy it into Adaptive Server.

Adaptive Server can accept data in any character or binary format, as long as the data file describes either the length of the fields or the **terminators**, the characters that separate columns.

The structures in the tables involved in the transfer need not be identical, because when **bcp**:

- Imports *from* a file, it appends data to an existing database table.
- Exports *to* a file, it overwrites the previous contents of the file.

When the transfer is complete, **bcp** reports the number of rows successfully copied and other performance information.

Requirements for Using *bcp*

Before using the **bcp** utility, you need to provide it with basic data information and prepare both the data for transfer and the command to access the data.

Basic Information Required

You must supply the following information to transfer data successfully to and from Adaptive Server:

- Name of the database and table or view
- Name of the operating system file
- Direction of the transfer (**in** or **out**)

In addition, you can use **bcp** to modify the storage type, storage length, and terminator for each column if you want to do so.

Permissions Needed to Copy Data

To use **bcp**, you must have an Adaptive Server account and the appropriate permissions on the database tables or views as well as the operating system files to use in the transfer.

- To copy data into a table, you must have **insert** permission on the table.
- To copy a table to an operating system file, you must have **select** permission on the following tables:
  - The table to copy
  - *sysobjects*
  - *syscolumns*
  - *sysindexes*

Pre-Transfer Task

Before you can use **bcp in**, you must prepare the command and the data for transfer:

- To use either fast or slow **bcp**, set **select into/bulkcopy/pllsort** to **true**. For example, to turn on this option for the *pubs2* database, you would enter:

```
sp_dboption pubs2, "select into/bulkcopy/pllsort", true
```

  For more information, see “bcp Modes” on page 16.

- To use fast **bcp**, remove indexes and triggers on the target table. For more information about this requirement, see “bcp Performance Issues” on page 17.

bcp Modes

bcp in works in one of two mode: fast or slow. Table 2-1 describes the differences between the two modes.

Table 2-1: Characteristics of fast and slow bcp

Type of *bcp*	Characteristics
Slow	Logs each row insert that it makes in a table. Used for tables that have one or more indexes or triggers.
Fast	Logs only page allocations. Copies data into tables without indexes or triggers at the fastest speed possible.

To determine the **bcp** mode that is best for your copying task, use the following information:

- The size of the table into which you are copying data,
- The amount of data that you are copying in,
- The number of indexes on the table, and
- The amount of spare database device space that you have for re-creating indexes.

Also, keep in mind that, although fast **bcp** might enhance performance, slow **bcp** gives you greater data recoverability.

bcp Performance Issues

Keeping indexes and triggers on a table causes the bulk copy utility to use slow **bcp** automatically. However, slow **bcp** can fill the transaction log very quickly.

- When you are copying a large number of rows, the performance penalty and log space requirements for using slow **bcp** can be severe.
- For extremely large tables, using slow **bcp** is not an option—its detailed log makes it much too slow.

To improve further the performance of **bcp**:

- Use partitioned tables. Several **bcp** sessions with a partitioned table can reduce dramatically the time required to copy the data. However, such performance improvements are more noticeable in fast **bcp** than in slow **bcp**.
- Use **bcp** in parallel to increase performance dramatically. Parallel bulk copy can provide balanced data distribution across partitions. For information about using parallel bulk copy, see "Using Parallel Bulk Copy to Copy Data Into a Specific Partition" on page 22.

Using Fast *bcp* or Slow *bcp*

The existence of indexes and triggers on tables does affect the speed of the transfer:

- When you copy in to a table that has indexes or triggers, **bcp** automatically uses its slow mode, which logs data inserts in the transaction log. These logged inserts can cause the transaction log to become very large.

  To control this data excess and ensure that the database is fully recoverable in the event of a failure, you can back up the log with **dump transaction**.

  Note **bcp** does not fire any trigger that exists on the target table.

- To save time, fast **bcp** does not log data inserts in the transaction log, but logs only the page allocations. For copying data in, **bcp** is fastest if your database table has no indexes or triggers.

However, if you used fast **bcp** to make data inserts, which fast **bcp** does not log, you cannot back up (**dump**) the transaction log to a device. The changes are not in the log, and a restore cannot recover non-existing backup data. The requested backup (**dump transaction**) produces an error message that instructs you to use **dump database** instead. This restriction remains in force until a **dump database** successfully completes.

Configuring the Database to Use Fast *bcp*

To allow a user to copy in data using fast **bcp**, either a System Administrator or the Database Owner first must use the **sp_dboption** system procedure to set **select into/bulkcopy/pllsort** to **true** on the database that contains the target table or tables. Should the option be **false** when a user tries to use fast **bcp** to copy data into a table that does not have indexes or triggers, Adaptive Server generates an error message.

Note You do not need to set the **select into/bulkcopy/pllsort** option to **true** to copy out data from or to copy in data to a table that has indexes or triggers. Slow **bcp** always copies tables with indexes or triggers and logs all inserts.

The following table shows which mode **bcp** automatically uses to copy in data and the possible settings for the **select into/bulkcopy/pllsort** option and indicates which combinations of mode and option allow the data transfer and backup.

Table 2-2: Fast and slow bcp and the select into/bulkcopy/pllsort option

select into/bulkcopy/pllsort	true / on	false / off
fast bcp on a target table that has no indexes or triggers	Transfer occurs. Cannot dump transaction to a device.	Cancels fast bcp automatically.
slow bcp on a target table that has one or more indexes or triggers	Transfer occurs. Allows dump transaction to occur.	Transfer occurs. Allows dump transaction to occur.

By default, the **select into/bulkcopy/pllsort** option is set to **false** (off) in newly created databases. To change the default setting for future databases, turn this option on (set to **true**) in the *model* database.

Dropping Indexes and Triggers

If you are copying a very large number of rows, you must have 1.2 times the amount of space needed for the data and enough space for the server to reconstruct a clustered index.

- If space is available, you can use **drop index** and **drop trigger** to drop all the indexes and triggers beforehand.

- If you don't have enough space for the server to sort the data and build the index or indexes, use slow **bcp.**

Steps for Copying In Data with Fast *bcp*

Table 2-3 summarizes the steps for copying in data to Adaptive Server using fast **bcp**.

Table 2-3: Steps for copying in data using fast bcp

Step	Who Can Do It
Use sp_dboption to set select into/bulkcopy/pllsort to true. Run checkpoint in the database that was changed.	System Administrator or Database Owner
Be sure that you have enough space to re-create any indexes and triggers on the table. Drop the indexes and triggers on the table.	Table owner
Be sure that you have insert permission on the table.	Granted by the table owner
Perform the copy with bcp.	Any user with *insert* permission
Re-create the indexes and triggers.	Table owner
Reset sp_dboption, if desired, and run checkpoint in the database that was changed.	System Administrator or Database Owner
Use dump database to back up the newly inserted data.	System Administrator, Operator, or Database Owner
Run stored procedures or queries to determine whether any of the newly loaded data violates rules.	Table owner or stored procedure owner

Bulk Copying Data into Partitioned Tables

In certain circumstances, you can improve **bcp** performance dramatically by executing several **bcp** sessions with a partitioned table.

Partitioned tables improve insert performance by reducing lock contention and by distributing I/O over multiple devices. **bcp** performance with partitioned tables is improved primarily because of this distributed I/O.

Guidelines

When you execute a **bcp** session on a partitioned table, consider the following:

- A partitioned table improves performance only when you are bulk copying *in* to the table.
- The performance of slow **bcp** does not improve as much with partitioned tables. Instead, drop all indexes and triggers and use fast **bcp**, as described in Table 2-3 on page 19, to increase performance.
- Network traffic can quickly become a bottleneck when multiple **bcp** sessions are being executed. If possible, use a local connection to the Adaptive Server to avoid this bottleneck.

Methods of Copying Data into a Partition

There are two methods for copying data into a partitioned heap table:

- Copying the data randomly without regard to the partition to which data is copied.
- Copying the data into a specific partition.

  If the table has a clustered index, **bcp** runs in slow mode and allows the index to control the placement of rows.

Copying Data Randomly into Partitions

To copy data randomly into partitioned tables when using multiple **bcp** sessions, you must perform the following actions:

1 Configure the table with as many partitions and physical devices as you require for your system.

  For more information, see the *Performance and Tuning Guide*, and "Using Parallel Bulk Copy to Copy Data Into a Specific Partition" on page 22 of this manual.

2 Make sure Adaptive Server is configured with enough locks to support multiple **bcp** sessions. For information on configuring locks, see the *System Administration Guide*.

3 Remove the triggers and indexes on the table and enable fast **bcp**.

To do so, follow the instructions in "Using Fast bcp or Slow bcp" on page 17.

Note If you use slow **bcp**, performance may not improve significantly after you remove the triggers and indexes. Also, if the table contains indexes, you may experience deadlocks on the index pages.

---

4 Divide the **bcp** input file into as many files of equal size as the number of planned simultaneous **bcp** sessions.

   You also can use the **-F***first_row* and **-L***last_row* options to specify the start and end of each "input file".

5 Execute the **bcp** sessions with separate files in parallel, preferrably on the local Adaptive Server machine.

   For example, on UNIX platforms, you can execute different sessions in different shell windows or start individual **bcp** sessions in the background.

For a detailed description of copying data into partitioned tables, see the *Performance and Tuning Guide.*

Monitor the *bcp* Session with *dbcc checktable* and *sp_helpsegment*

If you do not specify which partition the **bcp** sessions should use, Adaptive Server randomly assigns the multiple **bcp** sessions to the table's available partitions. If this random assignment occurs, be sure to monitor the partitions to ensure that the process has evenly distributed the inserts.

- Use the **dbcc checktable** command periodically to check the total page counts for each partition, or
- Use **sp_helpsegment** or **sp_helpartition**, which do not lock the database objects, to perform a similar check.

  For more information about **dbcc checktable**, see the *System Administration Guide*. For more information about **sp_helpsegment** and **sp_helpartition**, see the *Adaptive Server Reference Manual*.

  For more information about table partitions, see the *Performance and Tuning Guide*.

Reducing Logging by Increasing Page Allocations

If you are using fast **bcp**, consider that each **bcp in** batch requires the page manager to allocate one or more extents. Each such allocation generates a single log record.

Use the **number of preallocated extents** configuration parameter to specify how many extents Adaptive Server is to allocate through the page manager.

- Valid values for the **number of preallocated extents** configuration parameter are from 0 to 31; the default value is 2.
- You must restart Adaptive Server to change the value.
- When performing large **bcp** operations, increase this number to prevent the page allocations from filling the log.
- Set this value to 0 to prevent large extent allocations, so that the page manager performs only single page allocations.

Adaptive Server may allocate more pages than are actually needed, so keep the value small when space is limited. These pages are deallocated at the end of the batch.

For more information, see the *System Administration Guide*.

Using Parallel Bulk Copy to Copy Data Into a Specific Partition

Use parallel bulk copy to copy data in parallel to a specific partition. Parallel bulk copy substantially increases performance during **bcp** sessions because it can split large bulk copy jobs into multiple sessions and run the sessions concurrently.

To use parallel bulk copy:

- The destination table must be partitioned.
  - Use **sp_helpartition** to see the number of partitions on the table.
  - Use **alter table ... partition** to partition the table, if the table is not already partitioned.
- The destination table should not contain indexes because:
  - If the table has a clustered index, this index determines the physical placement of the data, causing the partition specification in the **bcp** command to be ignored.
  - If any indexes exist, **bcp** automatically uses its slow bulk copy instead of its fast bulk copy mode.
- If nonclustered indexes exist on the tables, parallel bulk copy is likely to lead to deadlocks on index pages.
- Each partition should reside on a separate physical disk for the best performance.

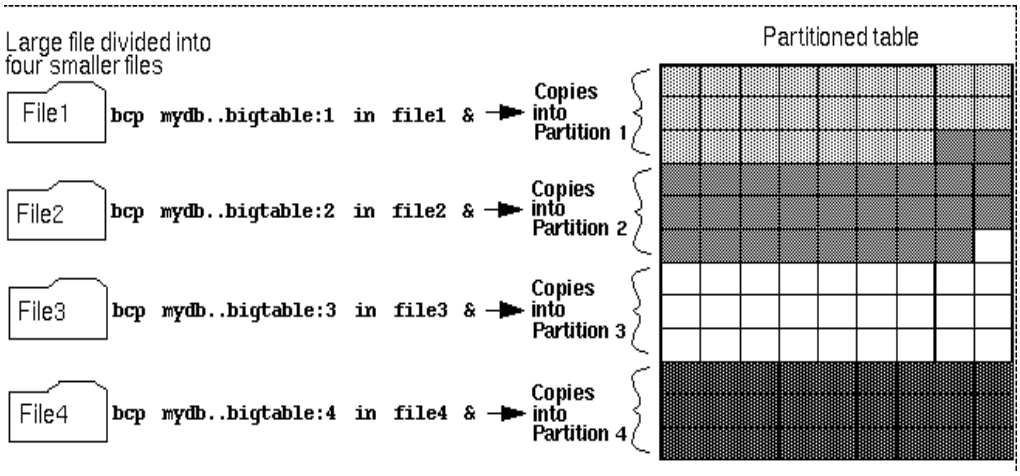
- Before you copy data into your database, you must partition the table destined to contain the data.
- Parallel bulk copy can copy in to a table from multiple operating system files. To do so, use the following syntax:

  ```
  bcp tablename :partition_number in file_name
  ```

  You can also use the **-F***first_row* and **-L***last_row* flags to designate the first and last row of the same host file for each multiple parallel bulk copy session.

Figure 2-1 illustrates the parallel bulk copy process:

Figure 2-1: Copying data into a partitioned table using parallel bulk copy



For information about partitioning a table, see the *Performance and Tuning Guide*.

Note When using parallel bulk copy to copy data out, you cannot specify which partitions **bcp** should use.

bcp in and Locks

When you copy in to a table using **bcp**, and particularly when you copy in to a table using parallel **bcp**, the copy process acquires the following locks:

- An exclusive intent lock on the table,

- An exclusive page lock on each data page, and
- An exclusive lock on index pages, if any indexes exist.

If you are copying in very large tables, and especially if you are using simultaneous copies into a partitioned table, this can involve a very large number of locks.

To avoid running out of locks:

- Increase the number of locks.
  - To estimate the number of locks needed, use the following formula:

```
# of simultaneous batches * (rows_per_batch / (2016/row_length))
```

  - To see the row length for a table, use the following query:

```
1> select maxlen
2> from sysindexes
3> where id = object_id("tablename") and (indid = 0 or indid = 1)
```

    See the *System Administration Guide* for more information about setting the number of locks.
- Use the -**b***batchsize* flag to copy smaller batches; the default batch size is 1000 rows.
- Run fewer batches concurrently.

Parallel Bulk Copy Methods

Use one of the following methods to copy in data using parallel bulk copy:

- Start multiple **bcp** sessions in the background, being sure to:
  - Specify the password at the command line.
  - Use native mode, character mode, or a format file.

  You can start **bcp** as many times as the table is partitioned.
- Create and use a format file:
  a Start **bcp** in interactive mode.
  b Answer the prompts.
  c Create a format file that stores your responses.
  d Put the process in the background when the copy begins.

e Issue the next **bcp** command, and specify the format file created with the first **bcp** command.

- Start **bcp** sessions in multiple windows.

Parallel Bulk Copy Syntax

The syntax for parallel bulk copy is:

```
bcp table_name[:partition_number] in file_name -Pmypassword
```

where:

- *table_name* is the name of the table into which you are copying the data,
- *partition_number* is the number of the partition into which you are copying,
- *file_name* is the host file that contains the data, and
- *mypassword* is your password.

Using Parallel Bulk Copy on Partitioned Tables

To copy sorted data in parallel into a specific partition:

- Specify the partition by appending a colon (:) plus the partition number to the table name. For example:

  ```
  publishers:10
  ```

  Note The partition you specify must exist before you issue the **bcp** command.

- Split the sorted data into separate files, or delineate the "files" by specifying the first row (**-F***first_row*) and the last row (**-L***last_row*) of the host file.
- Note the number of partitions in the table. This number limits the number of parallel bulk copy sessions that you can start.

  For example, if a table has four partitions, and you start five parallel bulk copy jobs, only the first four jobs can run in parallel; the fifth job does not start until one of the first four jobs finish.

bcp copies each file or set of line numbers to a separate partition. For example, to use parallel bulk copy to copy in sorted data to *mydb..bigtable* from four files into four partitions, enter:

```
bcp mydb..bigtable:1 in file1 -Pmypassword -c &
bcp mydb..bigtable:2 in file2 -Pmypassword -c &
bcp mydb..bigtable:3 in file3 -Pmypassword -c &
bcp mydb..bigtable:4 in file4 -Pmypassword -c &
```

Parallel Bulk Copy and IDENTITY Columns

When you are using parallel bulk copy, IDENTITY columns can cause a bottleneck. As **bcp** reads in the data, the utility both generates the values of the IDENTITY column and updates the IDENTITY column's maximum value for each row. This extra work may adversely affect the performance improvement that you expected to receive from using parallel bulk copy.

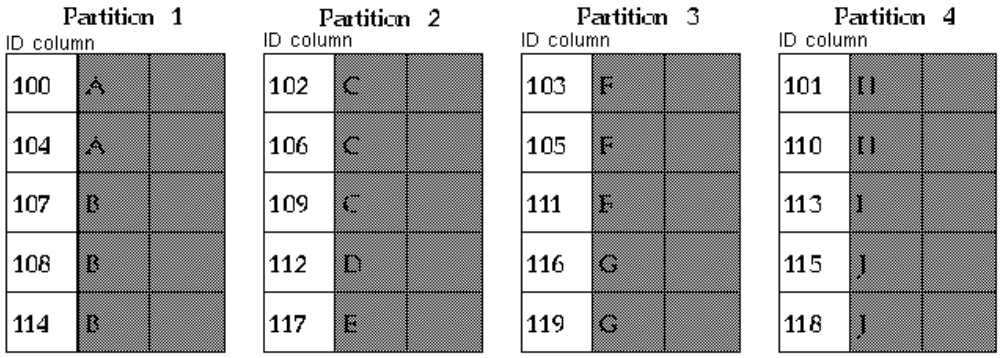
To avoid this bottleneck, you can explicitly specify the IDENTITY starting point for each session.

Retaining Sort Order

If you copy sorted data into the table without explicitly specifying the IDENTITY starting point, **bcp** might not generate the IDENTITY column values in sorted order. Parallel bulk copy reads the information into all the partitions simultaneously and updates the values of the IDENTITY column as it reads in the data.

A **bcp** statement with no explicit starting point would produce IDENTITY column numbers similar to those shown in Figure 2-2:

Figure 2-2: Producing IDENTITY columns in sorted order



The table has a maximum IDENTITY column number of 119, but the order is no longer meaningful.

If you want Adaptive Server to enforce unique IDENTITY column values, you must run **bcp** with either the **-g** or **-E** parameter.

Specifying the Starting Point from the Command Line

Use the **-g***id_start_value* flag to specify an IDENTITY starting point for a session in the command line.

The **-g** parameter instructs Adaptive Server to generate a sequence of IDENTITY column values for the **bcp** session without checking and updating the maximum value of the table's IDENTITY column for each row. Instead of checking, Adaptive Server updates the maximum value at the end of each batch.

Warning! You can create duplicate identity values inadvertently when you specify identity value ranges that overlap. For more information, see the text that follows.

To specify a starting IDENTITY value, enter:

```
bcp [-gid_start_value]
```

For example, to copy in four files, each of which has 100 rows, enter:

```
bcp mydb..bigtable in file1 -g100
bcp mydb..bigtable in file2 -g200
bcp mydb..bigtable in file3 -g300
bcp mydb..bigtable in file4 -g400
```

However, using the **-g** parameter does not guarantee that the IDENTITY column values are unique. To ensure uniqueness, you must:

- Know how many rows are in the input files and what the highest existing value is. Use this information to set the starting values with the **-g** parameter and generate ranges that do not overlap.

  In the example above, if any file contains more than 100 rows, the identity values overlap into the next 100 rows of data, creating duplicate identity values.

- Make sure that no one else is inserting data that can produce conflicting IDENTITY values.

Specifying the Starting Point Using the Data File

Use the **-E** parameter to set the IDENTITY starting point explicitly from the data file.

The -**E** parameter instructs **bcp** to read the value from the data file and send it to the server which inserts the value into the table. If the number of rows inserted exceeds the maximum possible IDENTITY column value, Adaptive Server returns an error.

Using the *bcp* Options

The information in this section clarifies some of the more complex options of the **bcp** syntax. For a complete description of the syntax, see **bcp on page 90**.

Using the Default Formats

When you use the native or character options, **bcp** operates non-interactively and only asks you for your Adaptive Server password.

Native Format

```
bcp pubs2..publishers out pub_out -n
```

Here are the contents of *pub_out*:

```
0736^MNew Age Books^FBoston^BMA0877^PBinnet  & Hardley^J
Washington^BDC1389^TAlgodata Infosystems^HBerkeley^BCA
```

bcp prefixed each field, except the *pub_id*, which is a char(4) datatype, with an ASCII character equivalent to the length of the data in the field. For example, "New Age Books" is 13 characters long, and ^M (Ctrl-M) is ASCII 13.

All the table data stored in the *pub_out* file is char or varchar data, so it is human-readable. In a table with numeric data, **bcp** writes the information to the file in the operating system's data representation format, which may not be human-readable.

Note The **bcp** utility does not support copying data in native format from different operating systems; for example, copying from NT to UNIX. Use the **-c** flag if you need to use **bcp** to copy files from one operating system to another.

Warning! Do not use row terminator (**-t)** or field terminator (**-r)** parameters with **bcp** in native format. Results are unpredictable and data could become corrupted.

Character Format

Character format (**-c**) uses the char datatype for all columns. It inserts tabs between fields in each row and a newline terminator at the end of each row.

For example, the following command copies out the data from the *publishers* table in character format to the file *pub_out*:

```
bcp pubs2..publishers out pub_out -c
```

The preceding command produces the following **bcp** output:

```
0736   New Age Books        Boston      MA
0877   Binnet  & Hardley     Washington  DC
1389   Algodata Infosystems Berkeley    CA
```

Changing Terminators From the Command Line

Terminators are the characters that separate data fields (field terminators). The row terminator is the field terminator of the last field in the table or file. Use the **-t***field_terminator* and **-r***row_terminator* command line options with the character format option (**-c**) to change the terminators from the command line.

The following example uses the comma (**,**) as the field terminator and Return (\r) as the row terminator.

```
bcp pubs2..publishers out pub_out -c -t , -r \\n
```

Remember to “escape” the backslash, if necessary, for your operating system command shell.

This **bcp** command line produces the following information:

```
0736,New Age Books,Boston,MA
0877,Binnet  & Hardley,Washington,DC
1389,Algodata Infosystems,Berkeley,CA
```

Note You can use the **-t** and **-r** options to change the default terminators without including the character option (**-c**).

Changing the Defaults: Interactive *bcp*

If you do not specify native (**-n**) or character (**-c**) format, **bcp** prompts you interactively for:

- The file storage type,
- The prefix length,
- The terminator for each column of data to be copied, and
- A field length for fields that are to be stored as char or binary.

The default values for these prompts produce the same results as using the native format and provide a simple means for copying data out of a database for later reloading into Adaptive Server.

If you are copying data to or from Adaptive Server for use with other programs, base your answers to the prompts on the format required by the other software.

These four prompts provide an extremely flexible system that allows you either to read a file from other software or to create a file that requires little or no editing to conform to many other data formats.

The following sections discuss these prompts and the way they interact to affect the data.

File Storage Type

The file storage type prompt offers you choices about how to store the data in the file. You can copy data into a file as:

- Its database table type,
- A character string, or
- Any datatype for which implicit conversion is supported.

Note **bcp** copies user-defined datatypes as their base types.

Table 2-4 shows the default storage type for each Adaptive Server datatype and the abbreviations that are acceptable to **bcp**.

- For the most compact storage, use the default value.
- For character files, use char.

- Keep in mind that the date storage type is the Adaptive Server internal storage format of datetime, not the host operating system format of the date.
- timestamp data is treated as binary(8).

In Table 2-4, brackets [] indicate that you can use the initial character or the beginning characters of the word. For example, for "bit" you can use "b," "bi," or "bit."

Table 2-4: File storage datatypes for bcp

Table Datatype	Storage Type
char, varchar	c[har]
text	T[ext]
int	i[nt]
smallint	s[mallint]
tinyint	t[inyint]
float	f[loat]
money	m[oney]
bit	b[it]
datetime	d[atetime]
binary, varbinary, timestamp	x
image	I[mage]
smalldatetime	D
real	r
smallmoney	M
numeric	n
decimal	e

To display this list while using **bcp** interactively, type a question mark (?) in response to the prompt "Enter the file storage type".

The suggested values that appear in the prompts are the defaults. Remember that your response determines how the data is stored in the output file; you need not indicate the column's type in the database table.

bcp fails if you enter a type that is not either implicitly convertible or char. For example, you may not be able to use smallint for int data (you may get overflow errors), but you can use int for smallint.

When storing noncharacter datatypes as their database types, **bcp** writes the data to the file in Adaptive Server's internal data representation format for the host operating system, rather than in human-readable form.

Prefix Length

By default, **bcp** precedes each field that has a variable storage length with a string of one or more bytes indicating the length of the field. This prefix enables the most compact file storage.

The default values in the prompts indicate the most efficient prefix length:

- For fixed-length fields, the prefix length should be 0.
- For fields of 255 bytes or less, the default prefix length is 1.
- For text or image datatypes, the default prefix length is 4.
- For binary and varbinary datatypes that are being converted to char storage types, the default prefix length is 2, since each byte of table data requires 2 bytes of file storage.
- For binary, varbinary, and image data, use even numbers for the prefix and length. This requirement maintains consistency with Adaptive Server, which stores data as an even number of hexadecimal digits.
- For any data column that permits null values, use a prefix length, other than 0, or a terminator to denote the length of each row's data. **bcp** considers such columns, including columns with integer datatypes that might ordinarily be considered fixed-length columns, to be of variable length.
- For data with no prefix before its column, use a prefix length of 0.

Unless you supply a terminator, **bcp** pads each stored field with spaces to the full length specified at the next prompt, "length".

Because prefix lengths consist of *native* format integers, the resulting host file contains non-printable characters. The nature of these characters could prevent you from printing the host file or from transmitting it through a communications program that cannot handle non-human-readable characters.

For more information about prefix lengths, see Table 2-7 on page 39.

Field Length

In almost all cases, use the **bcp** default value for the storage length while copying data out.

Note The terms “length” and “storage length” in this section refer to the operating system file, not to Adaptive Server field lengths.

- If you are creating a file to reload into Adaptive Server, the default prefixes and length keep the storage space needed to a minimum.
- If you are creating a human-readable file, the default length prevents the truncation of data or the creation of overflow errors that cause **bcp** to fail.

Because it is possible to change the default length by supplying another value, you must be familiar with the data to transfer. If you are copying character data in from other software, carefully examine the source file before choosing length values.

Note If the storage type is non-character, **bcp** stores the data in the operating system’s native data representation and does not prompt for a length.

When **bcp** converts non-character data to character storage, it suggests a default field length that is large enough to store the data without truncating datetime data or causing an overflow of numeric data.

- The default lengths are the number of bytes needed to display the longest value for the Adaptive Server datatype.

  Table 2-5 lists the default field lengths for data conversion to character storage:

Table 2-5: Default field lengths for non-character to character datatypes

Datatype	Default Size
int	12 bytes
smallint	6 bytes
tinyint	3 bytes
float	25 bytes
money	24 bytes
bit	1 byte
datetime	26 bytes
smalldatetime	26 bytes

Datatype	Default Size
real	25 bytes
smallmoney	24 bytes

- If you specify a field length that is too short for numeric data when copying data out, **bcp** prints an overflow message and does not copy the data.
- The default length for binary and varbinary fields is twice the length defined for the column, since each byte of the field requires 2 bytes of file storage.
- If you accept the default storage length, the actual amount of storage space allocated depends on whether or not you specify a prefix length and terminators.
  - If you specify a prefix length of 1, 2, or 4, **bcp** uses a storage space of the actual length of the data, plus the length of the prefix, plus any terminators.
  - If you specify a prefix length of 0 and no terminator, **bcp** allocates the maximum amount of space shown in the prompt, which is the maximum space that may be needed for the datatype in question. In other words, **bcp** treats the field as if it were fixed length to determine where one field ends and the next begins.

    For example, if the field is defined as varchar(30), **bcp** uses 30 bytes for each value, even if some of the values are only 1 character long.
- **bcp** does not know how large any one data value will be before copying all the data, so it always pads char datatypes to their full specified length.

Field and Row Terminators

A terminator can be used to mark the end of a column or ,row separating one from the next. The default is no terminator.

- Field terminators separate table columns.
- A row terminator is a field terminator for the last field in the row of the table or file.

Terminators are very useful for dealing with character data because you can choose human-readable terminators. The **bcp** character option, which uses tabs between each column with a newline terminator at the end of each row, is an example of using terminators that enhance the readability of a data file.

When you prepare data for use with other programs, and when you want to use **bcp** to prepare tabular data, supply your own terminators. The available terminators are:

- Tabs, indicated by `\t`
- New lines, indicated by `\n`
- Carriage returns, indicated by `\r`
- Backslash, indicated by `\`
- Null terminators (no visible terminator), indicated by `\0`
- Any printable character, for example, *, A, t, |
- Strings of up to 10 printable characters, including some or all of the terminators listed above (for example, **\t**, end, !!!!!!!!!!, and \t--\n)

Note Control characters (ASCII 0–25) cannot be printed.

Choosing Terminators

Be sure to choose terminators with patterns that do not appear in any of the data.

For example, using a tab terminator with a string of data that also contains a tab creates an ambiguity: Which tab represents the end of the string? **bcp** always looks for the first possible terminator, which in this case would be incorrect, since the first tab it would encounter would be the one that is part of the data string.

Data in native format can also conflict with terminators. Given a column that contains a 4-byte integer in native format, if the values of these integers are not strictly limited, it will be impossible to choose a terminator that is guaranteed not to appear inside the data. Use **bcp**'s native format option for data in native format.

Note The "no terminator" is different from a "null terminator," which is an invisible, but real, character.

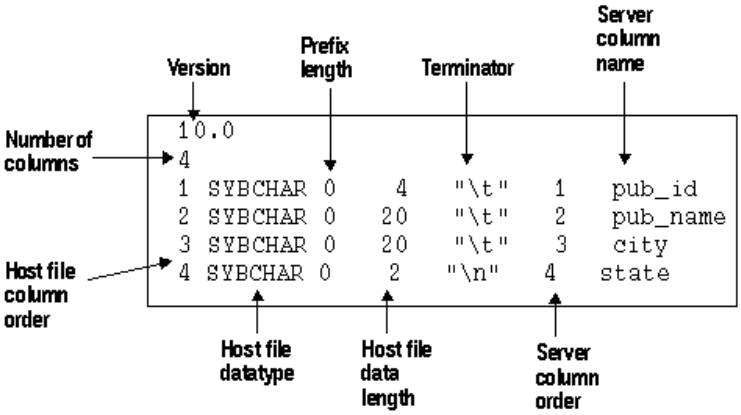
Using Format Files

After gathering information about each field in the table, **bcp** asks if you want to save the information to a **format file** and prompts for the file name.

Using a format file created for the data to be copied with the **bcp** utility allows you to copy data in or out non-interactively; that is, without being prompted by **bcp** for information. The format file supplies the information that **bcp** needs. You can use this newly created format file at any other time to copy the data back into Adaptive Server or to copy data out from the table.

Figure 2-3 illustrates the format of the **bcp** format files. It shows the *publishers* table from the *pubs2* database, with all the host file columns in character format, with no prefix, and using the default data length, a newline terminator at the end of the final column of a row, and tabs as terminators for all other columns.

Figure 2-3: bcp format file



Elements of the *bcp* Format File

The following list names the various elements of a **bcp** format file. Use Figure 2-3 on page 37 as the format file example.

- The **bcp** version is always the first line of the file. It specifies the version of **bcp** that you are using, not the Adaptive Server version, and appears is a literal string without quotation marks. In Figure 2-3, the version is 10.0.

- The second line of a **bcp** format file is the *number of columns*, which refers to the number of records in the format file, not including lines 1 and 2. Each column in the host table has one line.
- One line for each column follows the first and second lines in the database table. Each line consists of elements that are usually separated by tabs, except for the host file datatype and the prefix length which are usually separated by a space. These elements are:
  - Host file column order
  - Host file datatype
  - Prefix length
  - Host file data length
  - Terminator
  - Server column order
  - Server column name
  - Column precision
  - Column scale

The following sections describe the column elements in the format file.

Host File Column Order

The *host file column order* is the sequential number of the field in the host data file, which begins numbering at 1.

Host File Datatype

The *host file datatype* refers to the storage format of the field in the host data file, *not* the datatype of the database table column.

Table 2-6 lists the valid storage formats.

Table 2-6: Host file datatype storage format

Storage Format	Adaptive Server Datatype
SYBCHAR	*char/varchar* (ASCII)
SYBTEXT	text
SYBBINARY	binary
SYBIMAGE	image
SYBINT1	tinyint

Storage Format	Adaptive Server Datatype
SYBINT2	smallint
SYBINT4	int
SYBFLT8	float
SYBREAL	real
SYBBIT	bit
SYBNUMERIC	numeric
SYBDECIMAL	decimal
SYBMONEY	money
SYBMONEY4	smallmoney
SYBDATETIME	datetime
SYBDATETIME4	smalldatetime

Prefix Length

Prefix length indicates the number of bytes in the field length prefix. The length prefix is a 0-, 1-, 2-, or 4-byte unsigned integer value embedded in the host data file that specifies the actual length of data contained in the field. Some fields may have a length prefix while others do not.

Table 2-7 shows the allowable prefix length values.

Table 2-7: Allowable prefix length values

Length (in bytes)	Range
0	No prefix
1	2^8-1; 0–255
2	2^{16}-1; 0–65535
4	2^{32} -1; 0–4,294,967,295

Host File Data Length

Host file data length refers to the maximum number of bytes to copy for the field. To decide how much data to copy in or out, **bcp** uses either:

- The maximum field length,
- The prefix length, if any, or
- The field terminator string, if any.

If more than one method of field length specification is given, **bcp** chooses the one that copies the least amount of data.

Terminator

The *terminator* can be up to 30 bytes of characters enclosed in quotation marks (" "). The terminator designates the end of data for the host data file field.

Server Column Order

The *server column order* represents the *colid* (column ID) of the *syscolumns* column into which the host data file column is to be loaded. Together with the *host file column order*, this element maps host data file fields to the database table columns.

Server Column Name

The *server column name* is the name of the database table column into which this field is to be loaded.

Column Precision

The *column precision* is the precision of the database table column into which this field is to be loaded. This element is present only if the storage format is numeric or decimal.

Column Scale

The *column scale* is the scale of the database table column into which this field is to be loaded. This element is present only if the storage format is numeric or decimal.

Examples: Copying Out Data Interactively

By changing the default values of the prompts to **bcp**, you can prepare data for use with other software.

To create a human-readable file, respond to the **bcp** prompts as follows:

- File storage type, enter `0`
- Prefix length, enter `0`
- Field length, accept the default.
- Terminator – the field terminator you enter depends on the software that you plan to use.
  - Choose between delimited fields or fixed-length fields. Always use `\n`, the newline terminator, to terminate the last field.

    For fixed-length fields, do not use a terminator. Each field has a fixed length, with spaces to pad the fields. Adjacent fields, where the data completely fills the first field seem to run together, since there are no field separators on each line of output. See the example below.
  - For comma-delimited output, use a comma (`,`) as the terminator for each field. To create tabular output, use the tab character (`\t`).

Copying Out Data with Field Lengths

The following example uses fixed-length fields to create output in the personal computer format called SDF (system data format). This format can be easily read or produced by other software.

Note For information about format files, see "Using Format Files" on page 37.

```
bcp pubs2..sales out sal_out
```

The results as stored in the *sal_out* file are as follows:

```
5023   AB-123-DEF-425-1Z3        Oct 31 1985 12:00AM
5023   AB-872-DEF-732-2Z1        Nov 6 1985 12:00AM
5023   AX-532-FED-452-2Z7        Dec 1 1990 12:00AM
5023   BS-345-DSE-860-1F2        Dec 12 1986 12:00AM
5023   GH-542-NAD-713-9F9        Mar 15 1987 12:00AM
5023   NF-123-ADS-642-9G3        Jul 18 1987 12:00AM
5023   XS-135-DER-432-8J2        Mar 21 1991 12:00AM
```

```
5023   ZA-000-ASD-324-4D1  J      ul 27 1988 12:00AM
5023   ZD-123-DFG-752-9G8         Mar 21 1991 12:00AM
5023   ZS-645-CAT-415-1B2         Mar 21 1991 12:00AM
5023   ZZ-999-ZZZ-999-0A0         Mar 21 1991 12:00AM
6380   234518                     Sep 30 1987 12:00AM
6380   342157                     Dec 13 1985 12:00AM
6380   356921                     Feb 17 1991 12:00AM
7066   BA27618                    Oct 12 1985 12:00AM
7066   BA52498                    Oct 27 1987 12:00AM
7066   BA71224                    Aug 5 1988 12:00AM
7067   NB-1.142                   Jan 2 1987 12:00AM
7067   NB-3.142                   Jun 13 1990 12:00AM
7131   Asoap132                   Nov 16 1986 12:00AM
7131   Asoap432                   Dec 20 1990 12:00AM
7131   Fsoap867                   Sep 8 1987 12:00AM
7896   124152                     Aug 14 1986 12:00AM
7896   234518                     Feb 14 1991 12:00AM
8042   12-F-9                     Jul 13 1986 12:00AM
8042   13-E-7                     May 23 1989 12:00AM
8042   13-J-9                     Jan 13 1988 12:00AM
8042   55-V-7                     Mar 20 1991 12:00AM
8042   91-A-7                     Mar 20 1991 12:00AM
8042   91-V-7                     Mar 20 1991 12:00AM
```

The contents of the *sal_fmt* format file are as follows:

```
11.1
3
1  SYBCHAR 04  "" 1  stor_id
2  SYBCHAR 020 "" 2  ord_num
3  SYBCHAR 026 "" 3  date
```

For information about format files, see “Using Format Files” on page 37.

Copying Out Data with Delimiters

In the following examples, **bcp** copies data interactively from the *publishers* table to a file.

Note For information about format files, see “Using Format Files” on page 37.

Comma-delimited, Newline-delimited with Format File

The first example creates an output file with commas between all fields in a row and a newline terminator at the end of each row. This example creates a format file (*pub_fmt*) that you can use later to copy the same or similar data back into Adaptive Server.

```
bcp pubs2..publishers out pub_out
```

The results as stored in the *pub_out* file are as follows:

```
0736,New Age Books,Boston,MA
0877,Binnet  & Hardley,Washington,DC
1389,Algodata Infosystems,Berkeley,CA
```

The contents of the *pub_fmt* format file are as follows:

```
11.1
4
1  SYBCHAR 0 4  ","  1  pub_id
2  SYBCHAR 0 40 ","  2  pub_name
3  SYBCHAR 0 20 ","  3  city
4  SYBCHAR 0 2  "\n" 4  state
```

Tab-delimited with Format File

Similarly, the following example creates tab-delimited output from the table *pubs2..publishers* in the *pub_out* file.

```
bcp pubs2..publishers out pub_out
```

The results as stored in the *pub_out* file are as follows:

```
0736  New Age Books        Boston     MA
0877  Binnet  & Hardley     Washington DC
1389  Algodata Infosystems Berkeley   CA
```

The contents of the *pub_fmt* format file are as follows:

```
11.1
4
1  SYBCHAR 04   "\t"  1 pub_id
2  SYBCHAR 040  "\t"  2 pub_name
3  SYBCHAR 020  "\t"  3 city
4  SYBCHAR 02   "\n"  4 state
```

Examples: Copying In Data Interactively

To copy in data successfully to a table from a file, you must know what the terminators in the file are or what the field lengths are and specify them when you use **bcp**.

The following examples show how to copy data in, either with fixed field lengths or with delimiters, using **bcp** with or without a format file.

Copying In Data with Field Lengths

In this example, **bcp** copies data from the *salesnew* file into the *pubs2..sales* table.

In the *salesnew* file are three fields: the first is 4 characters long, the second is 20, and the third is 26 characters long. Each row ends with a newline terminator (\n), as follows:

```
5023ZS-731-AAB-780-2B9May 24 1993 12:00:00:000AM
5023XC-362-CFB-387-3Z5May 24 1993 12:00:00:000AM
6380837206       May 24 1993 12:00:00:000AM
6380838441       May 24 1993 12:00:00:000AM
```

Use the following command to copy in the data interactively from *salesnew*:

```
bcp pubs2..sales in salesnew
```

The system responds to the **bcp** command as follows:

```
Password:
Enter the file storage type of field stor_id [char]:
Enter prefix-length of field stor_id [0]:
Enter length of field stor_id [4]:
Enter field terminator [none]:
Enter the file storage type of field ord_num [char]:
Enter prefix-length of field ord_num [1]: 0
Enter length of field ord_num [20]:
Enter field terminator [none]:
Enter the file storage type of field date [datetime]: char
Enter prefix-length of field date [1]: 0
Enter length of field date [26]:
Enter field terminator [none]: \n
Do you want to save this format information in a file? [Y/n] y
Host filename [bcp.fmt]: salesin_fmt
Starting copy...
4 rows copied.
Clock Time (ms.): total = 1 Avg = 0 (116000.00 rows per sec.)
```

When you log in to Adaptive Server and access *sales*, you see the following data from *salesnew* appended to the table:

```
select * from sales
stor_idord_num            date
 ------- -------------------- ------------------------
--
 5023  AB-123-DEF-425-1Z3     Oct 31 1985 12:00AM
 5023  AB-872-DEF-732-2Z1     Nov  6 1985 12:00AM
 5023  AX-532-FED-452-2Z7     Dec  1 1990 12:00AM
 5023  BS-345-DSE-860-1F2     Dec 12 1986 12:00AM
 5023  GH-542-NAD-713-9F9     Mar 15 1987 12:00AM
 5023  NF-123-ADS-642-9G3     Jul 18 1987 12:00AM
 5023  XS-135-DER-432-8J2     Mar 21 1991 12:00AM
 5023  ZA-000-ASD-324-4D1     Jul 27 1988 12:00AM
 5023  ZD-123-DFG-752-9G8     Mar 21 1991 12:00AM
 5023  ZS-645-CAT-415-1B2     Mar 21 1991 12:00AM
 5023  ZZ-999-ZZZ-999-0A0     Mar 21 1991 12:00AM
 6380  234518                 Sep 30 1987 12:00AM
 6380  342157                 Dec 13 1985 12:00AM
 6380  356921                 Feb 17 1991 12:00AM
 7066  BA27618                Oct 12 1985 12:00AM
 7066  BA52498                Oct 27 1987 12:00AM
 7066  BA71224                Aug  5 1988 12:00AM
 7067  NB-1.142               Jan  2 1987 12:00AM
 7067  NB-3.142               Jun 13 1990 12:00AM
 7131  Asoap132               Nov 16 1986 12:00AM
 7131  Asoap432               Dec 20 1990 12:00AM
 7131  Fsoap867               Sep  8 1987 12:00AM
 7896  124152                 Aug 14 1986 12:00AM
 7896  234518                 Feb 14 1991 12:00AM
 8042  12-F-9                 Jul 13 1986 12:00AM
 8042  13-E-7                 May 23 1989 12:00AM
 8042  13-J-9                 Jan 13 1988 12:00AM
 8042  55-V-7                 Mar 20 1991 12:00AM
 8042  91-A-7                 Mar 20 1991 12:00AM
 8042  91-V-7                 Mar 20 1991 12:00AM
 (34 rows affected)
```

Since there is a unique clustered index on the *stor_id* and *ord_num* columns of *sales*, the new rows were sorted in order.

A conflict or violation can affect the copy process:

- Had there been any violations of the unique index on the columns in the data being copied from the file, **bcp** would have discarded the entire batch in which the violating row was encountered.

A batch size of 1 evaluates each row individually, but loads more slowly and creates a separate data page for each row during a fast **bcp** session.

- If the types copied in are incompatible with the database types, the entire copy fails.

Copying In Data with Delimiters

In the following example, **bcp** copies data from the file *newpubs* into the table *pubs2..publishers*. In the *newpubs* file, each field in a row ends with a tab character (\t) and each row ends with a newline terminator (\n), as follows:

```
1111 Stone Age Books    Boston    MA
2222Harley  & Davidson WashingtonDC
3333 Infodata Algosystems Berkeley  CA
```

Since *newpubs* contains all character data, you can use the character command line flag and specify the terminators with command line options as follows:

```
bcp pubs2..publishers in newpubs -c -t\\t -r\\n
```

Copying In Data with a Format File

To copy data back into Adaptive Server using the saved *pub_fmt* format file, run the following command:

```
bcp pubs2..publishers in pub_out -fpub_fmt
```

You can use the *pub_fmt* file to copy any data with the same format into Adaptive Server. If you have a similar data file with different delimiters, you can change the delimiters in the format file.

Similarly, you can edit the format file to reflect any changes to the field lengths, as long as all fields have the same length. For example, the *moresales* file contains the following:

```
804213-L-9 Jan 21 1993 12:00AM
804255-N-8 Mar 12 1993 12:00AM
804291-T-4 Mar 23 1993 12:00AM
804291-W-9 Mar 23 1993 12:00AM
```

Edit the *sal_fmt* format file to read as follows:

```
11.5
3
1  SYBCHAR 0 4 "" 1 stor_id
```

```
    2   SYBCHAR 0 7 "" 2 ord_num
    3   SYBCHAR 0 21 "\n" 3 date
```

Then enter the following command:

```
bcp pubs2..sales in moresales -fsal_fmt
```

The system responds as follows:

```
Starting copy...
4 rows copied.
 Clock Time (ms.): total = 1 Avg = 0 (116000.00 rows per sec.)
```

Using *bcp* with Alternate Languages

Adaptive Server stores data using its default character set, which is configured during installation. If your terminal does not support that default character set, it may send confusing characters to **bcp** when you respond to prompts either by typing or by using host file scripts.

Omitting all character set options causes **bcp** to use the character set that was named as the default for the platform. This default can cause communications problems:

- The default is not necessarily the same character set that was configured for Adaptive Server.
- The default may not necessarily be the character set that the client is using.

For more information about character sets and the associated flags, see "Configuring Client/Server Character Set Conversions" in the *System Administration Guide*.

Copy In and Batch Files

Batching applies only to bulk copying in; it has no effect when copying out. By default, Adaptive Server copies all the rows in batches of 1000 lines. To specify a different batch size, use the command line option (**-b***batch_size*).

When data is being copied in, it can be rejected by either Adaptive Server or **bcp**.

- Adaptive Server treats each batch as a separate transaction. If the server rejects any row in the batch, it rolls back the entire transaction.
- When **bcp** rejects a batch, it then continues to the next batch. Only fatal errors roll back the transaction.
- Adaptive Server generates error messages on a batch-by-batch basis, instead of row-by-row, and rejects each batch in which it finds an error. Error messages appear on your terminal and in the error file.
- **bcp** saves its error messages to an error file. A **bcp** stored error, for example, might be when Adaptive Server encounters a duplicate row for a table that has a unique index.

Improving Recoverability

To ensure better recoverability:

- Break large input files into smaller units.

  For example, if you use **bcp** with a batch size of 100,00 rows to bulk copy in 300,000 rows, and a fatal error occurs after row 200,000, **bcp** would have successfully copied in the first two batches—200,000 rows—to Adaptive Server. If you had not used batching, **bcp** would not have been able to copy in any rows to Adaptive Server.

- Set the **trunc log on chkpt** to **true** (on).

  The log entry for the transaction is available for truncation after the batch completes. If you copy into a database that has the **trunc log on chkpt** database option set on (**true**), the next automatic checkpoint removes the log entries for completed batches. This log cleaning breaks up large **bcp** operations and keeps the log from filling.

- Set **-b***batch_size* to **1**.

The batch size parameter set to 1 causes **bcp** to reject only the defective row. The error log from this setting allows you to identify exactly which row failed.

Note Because **bcp** creates 1 data page per batch, and setting **-b***batch_size* to 1 creates data pages with 1 row on each page, this setting causes the data to load slowly and takes up storage space.

Batches and Partitioned Tables

When you bulk copy data into a partitioned table without specifying a partition number, Adaptive Server randomly assigns each batch to an available partition. Copying rows in a single batch places all those rows in a single partition, which can lead to load imbalance in the partitioned table.

To help keep partitioned tables balanced, use a small batch size when bulk copying data or specify the partition ID during the **bcp** session. For information about partitioning tables, see the *Performance and Tuning Guide*.

Copy Out and *text* and *image* Data

When you copy out text or image data, Adaptive Server, by default, copies only the first 32K of data in a text or image field. The -T*text_or_image_size* parameter allows you to specify a different value. For example, if the text field to copy out contains up to 40K of data, you can use the following command to copy out all 40K:

```
bcp pubs2..publishers out -T40960
```

Note If a text or image field is larger than the given value or the default, **bcp** does not copy out the remaining data.

Specifying a Network Packet Size

To improve the performance of large bulk copy operations, you may want to use larger network packet sizes than the defaults. The -A*size* option specifies the network packet size to use for the **bcp** session that you are beginning.

The value of *size* must be:

- Between the values of the **default network packet size** and **max network packet size** configuration parameters, and
- A multiple of 512.

Note The new packet size remains in effect for the current **bcp** session only.

For example:

```
bcp pubs2..authors out -A 2048 -T40960
```

specifies that Adaptive Server send 40K of text or image data using a packet size of 2048 bytes for the **bcp** session.

Copy In and Error Files

When you specify the - **e** *error_file* option with copy in, **bcp** stores the rows that it cannot copy in to Adaptive Server in the specified error file.

- The error file stores the following elements:
  - A line that indicates which row failed and the error that occurred, and
  - A line that is an exact copy of the row in the host file.
- If the file name specified after -**e** already exists, **bcp** overwrites the existing file.
- If **bcp** does not encounter any errors, it does not create the file.

bcp in detects two types of errors:

- Data conversion errors
- Errors in building the row; for example, attempts to insert a NULL into columns that do not accept null values or to use invalid data formats, such as a 3-byte integer

The copy in process displays error messages on your monitor.

The following example loads the *newpubs* file into the *publishers* database, storing any error rows in the *pub_err* file:

```
bcp pubs2..publishers in newpubs -epub_err
```

Keep the following in mind when working with error files generated by copy in:

- **bcp** stores rows in an error file only when the **bcp** program itself detects the error.
- **bcp** continues to copy rows until **bcp** encounters the maximum number of error rows, at which point **bcp** stops the copy.
- **bcp** sends rows to Adaptive Server in batches, so **bcp** cannot save copies of rows that are rejected by Adaptive Server, for example, a duplicate row for a table that has a unique index.
- Adaptive Server generates error messages on a batch-by-batch basis, instead of row-by-row, and rejects the entire batch if it finds an error.
- It is not considered an error for Adaptive Server to reject duplicate rows if either **allow_dup_row** or **ignore_dup_key** was set when a table's index was created. The copy proceeds normally, but the duplicate rows are neither stored in the table nor in the **bcp** error file.

Copy Out and Error Files

During the copy out process, as with copy in, **bcp** overwrites any file of the same name and does not create an error file if no errors occurred.

There are two situations that cause rows to be logged in the error file during a copy out:

- A data conversion error in one of the row's columns
- An I/O error in writing to the host file

Keep the following in mind when working with error files generated by copy out:

- **bcp** logs rows in the error file in the default character format.
- All data values print as characters with tabs between the columns and a newline terminator at the end of each row.

Data Integrity: Defaults, Rules, and Triggers

To ensure integrity, **bcp** handles data to copy depending upon its element.

Defaults and Datatypes

When copying data into a table, **bcp** observes any defaults defined for the columns and datatypes. That is, if there is a null field in the data in a file, **bcp** loads the default value instead of the null value during the copy.

For example, here are two rows in a file to be loaded into *authors*:

```
409-56-7008,Bennet,David,415 658-9932,622 Pine St.,Berkeley,CA,USA,94705213-
46-8915,Green,Marjorie,,309 63rd St. #411,Oakland,CA,USA,94618
```

Commas separate the fields; a newline terminator separates the rows. Note that there is no phone number for Marjorie Green. Because the *phone* column of the *authors* table has a default of "unknown," the rows in the loaded table look like this:

```
409-56-7008  Bennet  David 415 658-9932  622 Pine St.
   Berkeley  CA  USA  94705
213-46-8915  Green   Marjorie  unknown       309 63rd St. #411
   Oakland   CA  USA  94618
```

Rules and Triggers

bcp, to enable its maximum speed for loading data**,** does not fire rules and triggers.

To find any rows that violate rules and triggers, copy the data into the table and run queries or stored procedures that test the rule or trigger conditions.

How *bcp* Differs from Other Utilities

The **bcp** utility, which copies entire tables or portions of a single table, is distinct from the other utilities that move data from one place to another.

The following list names these other utilities and their commands and describes how you can best use them to move data.

- The SQL commands **dump database**, **load database**, **dump transaction**, and **load transaction**.

  Use these commands for backup purposes only. Unlike **bcp**, the **dump** commands create a physical image of the entire database.

  You must use **load database** or **load transaction** to read data backed up with **dump database** or **dump transaction**.

  For information on using the SQL **dump** and **load** commands, see the *System Administration Guide* and the *Adaptive Server Reference Manual*.

- The data modification commands **insert**, **update**, and **delete**.

  Use these commands, respectively, to add new rows to, change existing rows in, or remove rows from a table or view.

  - Use the **insert** command with a **select** statement to move data between tables.
  - Use the **select** statement with an **into** clause to create a new table, based on 1) the columns named in the **select** statement, 2) the tables named in the **from** clause, and 3)data in the rows named in the **where** clause.

  For details on adding, changing, and deleting data, see **insert**, **update**, and **delete** in the *Adaptive Server Reference Manual*.

CHAPTER 3

Using *dsedit*

This chapter explains how to use the **dsedit** utility to edit the Adaptive Server interfaces file. Topics covered are:

Name	Page
Getting Started with dsedit	58
Opening an Editing Session	59
Adding, Viewing, and Editing Server Entries	61
Troubleshooting dsedit	64

Getting Started with *dsedit*

dsedit is an X-Windows-based graphical tool that lets you view and edit server entries in the interfaces file.

If your system does not have X-Windows, use **dscp** to configure server entries in the interfaces file. See "Getting Started with dscp" on page 68 for more information.

Starting *dsedit*

Before starting **dsedit**, make sure that you have UNIX write permission on the interfaces file.

If you are running **dsedit** from a remote machine, make sure that the DISPLAY environment variable is set so the **dsedit** screens will show on your machine instead of on the remote machine.

To set the DISPLAY environment variable:

1 Log in to the remote machine.

2 Enter the following command:

```
setenv DISPLAY your_machine_name:0.0
```

To start **dsedit**, enter:

```
$SYBASE/bin/dsedit
```

The Select a Directory Service screen appears. This screen lets you open editing sessions for the interfaces file. The full path name of the default interfaces file is shown in the Interfaces File to Edit box. The full path name of the configuration file is shown below it.

Opening an Editing Session

To open the default interfaces file for editing:

1 Select Sybase Interfaces File.

2 Click OK.

To open a file other than the default interfaces file:

1 Select Sybase Interfaces File.

2 Edit the displayed file name.

3 Click OK.

The Directory Service Session window appears.

You can open multiple interfaces file sessions with different files.

The Directory Service Session screen displays the full path name of the interfaces file and lists the server entries contained within it.

Add new server entry – displays the Server Entry Editor window, where you specify the name and network addresses for a new server entry.

Modify server entry – lets you view and modify the network addresses for a selected server entry. To view or modify a server entry, select the server in the list, then click Modify server entry to display the server's attributes in the Server Entry Editor window.

Copy server entry –lets you copy one or more entries to another interfaces file.

Close Session – closes the session window and writes changes to the interfaces file.

For procedures on using these buttons, see "Adding, Viewing, and Editing Server Entries" on page 61.

Clicking the Add new server entry or Modify server entry button in the Session screen displays the Server Entry Editor screen.

You use the Server Entry Editor window to view or edit server entries in an interfaces file.

Server name – if you are adding a server entry, type the name of the new server. If you are editing a server entry, you can edit the name field to rename the server. The new name cannot exist already in the interfaces file.

Available network transports – a list of the network addresses where the server accepts client connections. Do any of the following:

- To create a new address, click Add network transport. See “Adding, Viewing, and Editing Server Entries” on page 61.
- To edit an existing address, click Modify network transport. See “Adding, Viewing, and Editing Server Entries” on page 61.
- To remove a selected network address, click Delete network transport.
- To rearrange the order of addresses in the list, click Move network transport up or Move network transport down.

OK – Commits your changes and closes the window. Changes to the interfaces file are not applied until you close the session using the Close Session button in the Directory Service Session screen.

Cancel – Closes the window and discards any edits.

Adding, Viewing, and Editing Server Entries

To perform the procedures in this section, open the interfaces session window. For details, see "Opening an Editing Session" on page 59.

Note After performing each procedure in this section, you must click on *Close Session* to apply your edits to the interfaces file. Clicking this button also closes the interfaces session window.

To add a new server entry:

1 Click on Add new server entry.

The Server Entry Editor window appears.

2 Specify the name and network addresses for a new server entry.

To view or modify a server entry:

1 Click on Modify server entry.

The Server Entry Editor window displays the server's attributes.

2 Modify the attributes as desired.

To copy a server entry to another interfaces file:

1 Use one of the following methods to select the entries to copy:

- To copy a single entry, click it once.
- To copy a range of consecutive entries, click the first entry in the range, press and hold down Shift, and click the last entry in the range. (You can also select "backwards" by clicking the last entry, holding down Shift, and clicking the first entry.
- To select multiple, nonconsecutive entries, press and hold down the Ctrl key while you click each entry.

2 Click Copy server entry.

3 Select the Sybase interfaces file from the list.

4 Edit the displayed file name.

5 Click OK.

Adding or Editing Network Transport Addresses

The Network Transport Editor window allows you to view, edit, or create the transport addresses at which a server accepts client connections. This window displays the name of the server entry for the address and allows you to configure the following items:

- *Transport type* – specifies the protocol and interface for the address. For all platforms except Digital UNIX, values are **tcp**, **tli tcp**, **tli spx**, and **spx**. For Digital UNIX, values are **decnet**, **tcp**, and **tli tcp**.
- *Address information* – depending on the transport type, different address components are required. The following sections discuss address formats in detail.

TCP/IP Addresses

The address information for a TCP/IP entry consists of a host name (or IP address) and a port number (entered as a decimal number). For **tli tcp**-formatted interfaces entries, the host's IP address and the port number are converted to the 16-byte hexadecimal representation required for **tli tcp**-formatted interfaces entries.

In interfaces entries, use **tli tcp** for:

- All pre-10.0 clients on platforms that use **tli**-formatted interfaces entries
- Adaptive Server or Replication Server release 11.0.x or earlier on platforms that use **tli**-formatted interfaces entries

Use **tcp** for other clients and servers.

To indicate a TCP/IP address choose **tcp** or **tli tcp** from the Transport Type menu.

SPX/IPX Addresses

SPX/IPX addresses allow Adaptive Server to listen for connections from client applications running on a Novell network. SPX/IPX addresses consist of the following information:

- *Host address* – an eight-digit hexadecimal value representing the IP address of the computer on which the server runs. Each component of the dot-separated decimal IP address format maps to 1 byte in the hex address format. For example, if your host's IP address is 128.15.15.14, enter "800F0F0E" as the SPX/IPX host address value.
- *Port number* – the port number, expressed as a 4-digit hexadecimal number.
- *Endpoint* – the path for the device file that points to the SPX device driver. Defaults to */dev/mspx* on Solaris and */dev/nspx* on any other platform. If necessary, adjust the path so that it is correct for the machine on which the server runs. The default path is based on the platform on which you are running **dsedit**.

To indicate an SPX/IPX address choose **tli spx** or **spx** from the Transport Type menu.

Troubleshooting *dsedit*

This section lists some common problems and describes how to correct them.

The *dsedit* Utility Does Not Start

Check for the following:

- The SYBASE environment variable is not set or points to the wrong directory.
- X-Windows is not configured correctly. If you are running **dsedit** on a remote host, make sure that X-Windows clients on the remote host can connect to the X-Windows server on your own machine. See your X-Windows documentation for more troubleshooting information. If X-Windows is not available, use **dscp** instead of **dsedit**.

Error Message: “Unable to open X display”

dsedit might not work if the display machine is set up to reject X-Windows connections from remote hosts. If this is the problem, you will see a message similar to the following:

```
Unable to open X display. Check the value of your
$DISPLAY variable. If it is set correctly, use the
'xhost +' command on the display machine to authorize
use of the X display. If no X display is available, run
dscp instead of dsedit.
```

This error may be caused by either of the following situations:

- The value for the DISPLAY environment variable is not entered correctly or is not set.

  Solution: Enter the DISPLAY environment variable correctly.

- You are not authorized to open windows on the machine to which DISPLAY refers.

  Solution: Run the command **‘xhost +’** on the display machine.

Cannot Add, Modify, or Delete Server Entries

Check for permissions problems with the interfaces file. To edit interfaces entries, you must have write permission on both the interfaces file and the Sybase installation directory.

CHAPTER 4

Using *dscp*

dscp is a utility program that you use to view and edit server entries in the interfaces file. This section covers the following topics:

Name	Page
Getting Started with dscp	68
Working with Server Entries	70
Exiting dscp	76
Quick Reference for dscp Utility Commands	77

Getting Started with *dscp*

Starting *dscp*

To start **dscp**, enter:

```
$SYBASE/bin/dscp
```

The **dscp** prompt, `>>`, appears.

Getting Help with *dscp*

To view the **dscp** help screen, enter one of the following commands:

```
help
h
?
```

Using a *dscp* Session

Before you can view, add, or modify server entries, you must open a session so that you can interact with the interfaces file.

You can have multiple sessions open at one time.

Opening a Session

To open a session with the interfaces file, enter:

```
open InterfacesDriver
```

When you open a session, **dscp** provides the session's number. For example, if you open a session using the **open InterfacesDriver** command, **dscp** displays the following message:

```
ok

Session 1 InterfacesDriver>>
```

Listing Sessions

To list all open sessions, enter:

```
sess
```

Switching Between Open Sessions

To switch to another open session, enter:

```
switch sess
```

where *sess* is the session number.

For example, if you enter:

```
switch 3
```

you are switched to session 3. The **switch** keyword is optional. For example, entering:

```
3
```

also switches you to session 3.

Closing a Session

To close a session, enter:

```
close sess
```

where *sess* is the session number.

For example, if you enter:

```
close 3
```

session 3 is closed.

If you do not specify a session number, **dscp** closes the current session.

Working with Server Entries

Adding and Modifying Server Entries

After you open a session, you can add or modify server entries associated with that session.

Note When you add or modify a server entry, **dscp** automatically creates or modifies both master and query lines. The master line and the query line of an interfaces file entry contain identical information.

Each server entry is made up of a set of attributes. When you add or modify a server entry, **dscp** prompts you for information about each attribute. Table 4-2 describes each attribute:

Table 4-1: Server attributes

Attributes	Type of Value	Default Value and Valid Values	Can be edited when adding or modifying a server entry
Server Object Version	Integer	110	Adding: No Modifying: No
Server Name	Character string	n/a	Adding: n/a Modifying: No
Server Service	Character string	SQL SERVER	Adding: Yes Modifying: No
Server Status	Integer	4 Valid values are: 1 Active 2 Stopped 3 Failed 4 Unknown	Adding: No Modifying: No
Transport Type	Character string	tcp Valid values are: decnet, spx, tcp, tli, spx, tli tcp	Adding: Yes Modifying: Yes

Attributes	Type of Value	Default Value and Valid Values	Can be edited when adding or modifying a server entry
Transport Address	Character string	None Valid values are character strings recognized by the specified transport type.	Adding: Yes Modifying: Yes
Security Mechanism	Character string **Note** You can add up to 20 security mechanism strings for each server entry.	None Valid values are character strings associated with object identifiers defined in the user's *objectid.dat*.	Adding: Yes Modifying: Yes

Adding Server Entries

1 To add a server entry, enter:

```
add servername
```

You are now in *add mode*. You can continue to add server entries, but you cannot execute any other **dscp** commands until you exit this mode. While in add mode, **dscp** prompts you for information about *servername*.

2 Do one of the following:

- Enter a value for each attribute, or
- Press Return to accept the default value, which is shown in brackets [ ].

For example, if you enter:

```
add myserver
```

dscp prompts for the following information:

```
Service: [SQL Server]
Transport Type: [tcp] tcp
Transport Address: victory 8001
Security Mechanism []:
```

A server entry can have up to 20 transport type/address combinations associated with it.

For a description of the server attributes, see Table 4-1 on page 70.

1 To exit add mode, enter:

```
#done
```

Modifying Server Entries

You cannot use **dscp** to modify the Version, Service, and Status entries in the *interfaces* file.

1 To modify a server entry, enter:

```
mod servername
```

You are now in *modify mode*. You can continue to modify server entries, but you cannot execute any other **dscp** commands until you exit this mode. In modify mode, **dscp** prompts you for information about *servername*.

2 Do one of the following:

- Enter a value for each attribute, or
- Press Return to accept the default value, which is shown in brackets [ ].

For example, if you enter:

```
mod myserver
```

dscp prompts for the following information:

```
Version: [1]
Service: [SQL Server] Open Server
Status: [4]
Address:
Transport Type: [tcp]
Transport Address: [victory 1824] victory 1826
Transport Type: [tcp]
Transport Address: [victory 1828]
Transport Type: []
Security Mechanism []:
```

For a description of the server attributes, see Table 4-1 on page 70.

1 To delete an address, enter:

```
#del
```

2 To exit modify mode, enter:

```
#done
```

Copying Server Entries

dscp allows you to copy server entries within a session and between two sessions. You have four options when copying a server entry.

You can copy:

- A server entry to a new name in the current session
- A server entry to a different session
- A server entry to a new name in a different session
- All entries in the current session to a different session

Creating New Server Entries Within a Session By Copying

To create a new server entry within a session by copying, enter:

```
copy name1 to name2
```

For example, if you enter:

```
copy myserver to my_server
```

dscp creates a new entry, "my_server," that is identical to "myserver." You can then modify the new entry and leave the original intact.

Copying Entries Between Sessions

- To copy a server entry without changing the name, enter:

  ```
  copy name1 to sess
  ```

  For example, if you enter:

  ```
  copy myserver to 2
  ```

  dscp copies the "myserver" entry in the current session to session 2.

- To copy a server entry and rename it, enter:

  ```
  copy name1 to sess name2
  ```

  For example, if you enter:

  ```
  copy myserver to 2 my_server
  ```

dscp copies the “myserver” entry in the current session to session 2 and renames it “my_server.”

Copying All Entries to a Different Session

To copy all entries in the current session to a different session, enter:

```
copyall sess
```

For example, if you enter:

```
copyall 2
```

dscp copies all entries in the current session to session 2.

Listing Server Entries

- To list the names of server entries associated with a session, enter:

  ```
  list
  ```

- To list the attributes of server entries associated with a session, enter:

  ```
  list all
  ```

For a description of server attributes, see Table 4-1 on page 70.

Viewing Contents of Server Entries

To view the contents of a server entry, enter:

```
read servername
```

For example, if you enter:

```
read myserver
```

the following information is displayed:

```
DIT base for object: interfaces
Distinguish name: myserver
Server Version: 1
Server Name: myserver
Server Service: SQL Server
Server Status: 4 (Unknown)
Server Address:
Transport Type: tcp
```

```
Transport Addr: victory 1824
Transport Type: tcp
Transport Addr: victory 1828
```

For a description of the server attributes, see Table 4-1 on page 70.

Deleting Server Entries

You can delete one entry or all entries associated with a session.

- To delete one server entry, enter:

  ```
  del servername
  ```

  For example, if you enter:

  ```
  del myserver
  ```

  dscp deletes the entry for “myserver.”

- To delete all entries associated with a session, enter:

  ```
  delete-all
  ```

Exiting *dscp*

To exit **dscp**, enter one of the following commands:

```
exit
quit
```

Quick Reference for *dscp* Utility Commands

dscp allows you to perform functions by entering commands at the **dscp** prompt. Table 4-2 provides a quick reference to these commands.

Table 4-2: dscp commands

Command	Description
add *servername*	Adds server entry *servername* in the current session. dscp prompts you for information about *servername*. Press Return to accept the default value, which is shown in square brackets [ ]. Enter "#done" to exit add mode.
addattr *servername*	Adds an attribute to the server entry *servername* in the current session.
close [*sess*]	Closes a session identified by the *sess* number. If you do not specify *sess*, closes the current session.
config	Displays configuration information related to your Sybase environment.
copy *name1* to {*name2* | *sess* | *sess name2*}	Copies server entry *name1* in the current session to: • Server entry *name2* in the current session, • Session *sess*, or • Server entry *name2 in session sess*.
copyall to *sess*	Copies all server entries in the current session to session *sess*.
del *servername*	Deletes server entry *servername* in the current session.
delete-all	Deletes all server entries in the current session.
exit	Exits dscp.
help, ?, h	Displays the on-line help.
list [all]	Lists the server entries for the current session. To list the names of the entries, use the list command. To list the attributes for each entry, use the list all command.
mod *servername*	Modifies server entry *servername* in the current session. dscp prompts you for information about *servername*. Press Return to accept the default value, which is shown in square brackets [ ]. Enter "#done" to exit modify mode.
open [*dsname*]	Opens a session for the specified directory service, where dsname is the directory service name. If you do not specify a value for *dsname*, this command opens a session for the default directory service. To open a session, specify the value "InterfacesDriver" for *dsname*.
quit	Exits dscp.
read *servername*	Displays the contents of server entry *servername*.
sess	Lists all open sessions.
[switch] *sess*	Makes session number *sess* the current session.

CHAPTER 5

Utility Commands Reference

This chapter contains reference pages for the following utility program commands:

Name	Page
backupserver	85
bcp	90
bcp_dce	105
buildmaster	122
charset	125
dataserver	126
defncopy	129
defncopy_dce	134
dscp	139
dscp_dce	140
dsedit	141
dsedit_dce	142
extractjava	143
installjava	146
isql_dce	158
langinstall	166
optdiag	168
showserver	175
sqlloc	176
sqllocres	177
sqlupgrade	178
sqlupgraderes	179
srvbuild	180
srvbuildres	181
startserver	182
sybload	185
sybsetup	187

Starting A Utility Program

You enter a utility program command at the system prompt in a UNIX shell.

Place characters with special meaning to the shell, such as the backslash (\), asterisk (*), slash (/), and spaces, in quotes. You can precede some special characters with the backslash (\) to "escape" them. This prevents the shell from interpreting the special characters.

Table 5-1 describes the utility programs available with Adaptive Server for UNIX platforms.

Note The utility programs described in Table 5-1 may allow you to use a **-P** parameter to enter your password. If security is an issue, do not use this parameter to specify your password. Another user may have an opportunity to see it. Instead, log in as usual without the **-P** parameter, and let Adaptive Server prompt you for your password.

Table 5-1: Utility programs for UNIX

Utility	Description
backupserver	Executable form of the Backup Server™ program.
bcp	Copies rows in a database table to or from an operating system file in a user-specified format.
buildmaster	Builds the master device and creates the *master*, *model*, and *tempdb* databases on the device.
charset	Loads the character sets and sort order files.
dataserver	Executable form of the Adaptive Server program.
defncopy	Copies definitions for specified views, rules, defaults, triggers, procedures, or reports from a database to an operating system file or from an operating system file to a database.
dscp	Allows you to view and edit server entries in the interfaces file in command line mode.
dscp_dce	Creates and modifies network connection information in the interfaces file.
dsedit	Allows you to view and edit server entries in the interfaces file using a graphical user interface based on X11/Motif.
extractjava	Copies a retained JAR from an Adaptive Server to a client file.
installjava	Installs a JAR from a client file into an Adaptive Server.
isql	Interactive SQL parser to Adaptive Server.
langinstall	Installs a new language on the Adaptive Server.
showserver	Shows Adaptive Servers and Backup Servers that are currently running on the local machine.

Utility	Description
sqlloc	Installs and modifies languages, character sets, and sort order defaults for Adaptive Server in GUI mode.
sqllocres	Installs and modifies languages, character sets, and sort order defaults for Adaptive Server in command line mode.
sqlupgrade	Upgrades your currently installed release of Adaptive Server to the newest release in GUI mode.
sqlupgraderes	Upgrades your currently installed release of Adaptive Server to the newest release in command line mode.
srvbuild	Creates a new Adaptive Server, Backup Server, Monitor Server, or XP Server in GUI mode with default or user-specified values for key configuration attributes.
srvbuildres	Creates a new Adaptive Server, Backup Server, Monitor Server, or XP Server in command line mode with default or user-specified values for key configuration attributes.
startserver	Starts an Adaptive Server or a Backup Server.
sybload	Uploads Sybase products from the distribution media and builds the Sybase installation directory from the command line.
sybsetup	Installs and configures Adaptive Server from a single location using a GUI interface.

Utilities Quick Reference

This section provides a quick reference for the utilities, divided into the following categories:

- Installation or Configuration Utilities
- Utilities for Languages, Character Sets and Sort Orders
- Utilities to Start Servers
- Database Creation and Manipulation Utilities
- Utilities to Gather Information

Installation or Configuration Utilities

Use the following utilities to install or configure databases:

- **dscp** – allows you to view and edit server entries in the interfaces file from the command line.
- **dsedit** – allows you to view and edit server entries in the interfaces file using a graphical user interface based on X11/Motif.
- **sqlupgrade** – upgrades your currently installed release of Adaptive Server to the newest release using a graphical user interface based on X11/Motif.
- **sqlupgraderes** – upgrades your currently installed release of Adaptive Server to the newest release using resource files.
- **srvbuild** – creates a new Adaptive Server, Backup Server, Monitor Server, or XP Server with default or user-specified values for key configuration attributes using a graphical user interface based on X11/Motif.
- **srvbuildres** – creates a new Adaptive Server, Backup Server, Monitor Server, or XP Server, using resource files to specify values for key configuration attributes.
- **sybload** – uploads Sybase products from the distribution media and builds the Sybase installation directory.
- **sybsetup** – installs and configures Adaptive Server from a single location using a graphical user interface based on X11/Motif.

Utilities for Languages, Character Sets and Sort Orders

Use the following utilities to set languages, character sets and sort orders:

- **charset** – loads the character sets and sort order files.
- **langinstall** – installs a new language on an Adaptive Server.
- **sqlloc** – installs and modifies languages, character sets, and sort order defaults for Adaptive Server, using a graphical user interface based on X11/Motif.
- **sqllocres** – installs and modifies languages, character sets, and sort order defaults for Adaptive Server, using a resource file.

Utilities to Start Servers

Use the following utilities to start servers manually:

- **backupserver** – starts the Backup Server executable. Use the **startserver** command instead of this utility to start Backup Server manually.
- **dataserver** – starts the Adaptive Server executable. Use the **startserver** command instead of this utility to start Adaptive Server manually.
- **histserver**– starts the Historical Server executable. Use the **histserver** command instead of this utility to start Historical Server manually.
- **monserver**– starts the Monitor Server executable. Use the **monserver** command instead of this utility to start Monitor Server manually.
- **startserver** – starts an Adaptive Server and a Backup Server.

Database Creation and Manipulation Utilities

Use the following utilities to create and manipulate databases:

- **bcp** – copies a database table to or from an operating system file in a user-specified format.
- **buildmaster** – builds the master device and creates the *master*, *model*, and *tempdb* databases on the device.

- **defncopy** – copies definitions for specified views, rules, defaults, triggers, or procedures from a database to an operating system file or from an operating system file to a database.
- **extractjava** – copies a retained JAR and the classes it contains from an Adaptive Server to a client file.
- **installjava –** installs a JAR from a client file into an Adaptive Server database.
- **isql** – interactive SQL parser to Adaptive Server.
- **optdiag** – displays optimizer statistics or loads updated statistics into system table.

Utilities to Gather Information

Use the following utilities to gather information:

- **showserver** – shows the Adaptive Servers and Backup Servers that are currently running on the local machine.

backupserver

Description

The executable form of the Backup Server program.

Syntax

backupserver
[-C*server_connections*]
[-S*b_servername*]
[-I*interfaces_file*]
[-e*error_log_file*]
[-M*sybmultbuf_binary*]
[-N*network_connections*]
[-T*trace_value*]
[-L*Sybase_language_name*]
[-J*Sybase_character_set_name*]
[-P*active_service_threads*]
[-c*tape_config_file*]
[-V*level_number*]

or

backupserver -v

Parameters

-**C***server_connections*
specifies the number of server connections for the Backup Server. The Backup Server requires:

- Two connections for each dump session.
- One connection for each load session.
- One connection for volume change messages.

Allow a maximum of three times the number of expected concurrent dump and load sessions. The default value is 30 server connections.

-**S***b_servername*
specifies the name of the Backup Server to start. The default is SYB_BACKUP. This entry must specify the name of a Backup Server in the interfaces file.

-**I***interfaces_file*
specifies the name and location of the interfaces file to search when connecting to Backup Server. If **-I** is omitted, **backupserver** looks for a file named *interfaces* in the directory pointed to by your SYBASE environment variable.

-**e***error_log_file*
specifies the name and location of the Backup Server error log file used to report Open Server™ internal errors, **sybmultbuf** errors, errors that halt the Backup Server, and errors for disconnected sessions. All other errors are sent to the **notify** destination specified in the **dump database**, **dump transaction**, **load database**, and **load transaction** commands.

-**M***sybmultbuf_binary*
specifies the full path name of the **sybmultbuf** executable. Use this parameter only when starting Backup Server from a directory other than the *bin* directory of the Sybase installation directory, or when using a diagnostic version of **sybmultbuf**.

-**N***network_connections*
specifies the number of total network connections (DBPROCESSes) that the master Backup Server can originate. The default value is 25.

-**T***trace_value*
interprets *trace_value* as a bit mask (base-2 number). The 1 bits in *trace_value* correspond to Open Server Trace flags to turn on. If you specify more than one **-T** parameter on the command line, the final **-T** value overrides the values from earlier **-T** parameters. The *trace_value* must be a positive integer.

-**L***Sybase_language_name*
specifies the default language for Backup Server. If not specified, Backup Server uses the locale specified by the LC_ALL or LANG environment variables. If these variables are not set, Backup Server searches for the "default" entry in *locales.dat*.

Note The **-L** parameter does not override the value set in the LANG environment variable.

-**J***Sybase_character_set_name*
specifies the default character set for Backup Server.

-P*active_service_threads*
allows you to increase the number of stripes during multiple dump/load operations (with a maximum of 32 stripes per single operation). The default system-wide limit is 48, which includes all dumps and loads.

-c*tape_config_file*
specifies the name and location of the tape configuration file to search for tape device configuration information before doing a **dump database** or a **dump transaction**. If you do not specify **-c**, the default path name for the tape configuration file is *$SYBASE/backup_tape.cfg*.

-v
prints the version number and copyright message of the **backupserver** software and then exits.

-**V***level_number*
limits the messages that are printed to the Backup Server error log. The *level_number* variable determines the degree of error verbosity (-V) for Backup Server:

- **-V3** – displays only completion messages from a normal **dump** or **load** command and the following types of messages:

  Error messages from Backup Server and **sybmultbuf**

  Other **sybmultbuf** messages

  Volume change messages

  Open Server™ messages

  Trace print messages

  Informational messages from the System & Tape Auto Config modules

- **-V2** – displays:

  All **-V3** messages *plus*

  File creation and file mount messages

- **-V1** – displays:

  All **-V2** messages *plus*

  Phase messages

- **-V0** (default) – displays:

  All messages, including backup progress

  This limitation does not involve the messages that are sent to the client or console as determined by the **NOTIFY=** parameter in a **dump** or **load** command.

  This option also does not affect logging for the following message types:

- Open Server messages,
- Trace printing messages from **bs_traceprint**, and
- **sybmultbuf** messages

Usage

- Start Backup Server with the **startserver** command rather than by directly executing the **backupserver** program. To change default values, edit the *RUN_servername* file in your Sybase installation directory. See the **startserver** reference page for details.

- If you do not specify a Backup Server name with the **-S** parameter, and you have not set the environment variable DSLISTEN, **backupserver** uses the default Backup Server name SYB_BACKUP. The value of the DSLISTEN environment variable overrides this default value, and the **-S** parameter overrides both the default and the value specified in DSLISTEN.
- Whenever possible, the Backup Server and any Adaptive Servers that dump or load directly through the Backup Server should share the same interfaces file. The interfaces file that Backup Server uses must contain entries for:
  - Backup Server
  - Any other Backup Servers with which this Backup Server communicates
- Trace flags cause the Backup Server to print information regarding its operation while it is running, for debugging problems in the Backup Server. See the *Open Server Server-Library/C Reference Manual* for more details on trace flags. In UNIX, the Backup Server does not support use of the Open Server-defined SRV__TR symbols for **-T**.
- If Backup Server cannot find the *locales* and *charsets* directories specified by the **-L** and **-J** parameters, or if these parameters specify an incorrect language and character set combination, Backup Server issues an error message and uses the default language and character set.

See also

Utility	**startserver**

bcp

Description

Copies a database table to or from an operating system file in a user-specified format.

Syntax

bcp [[*database_name*.]*owner*.][*view_name* | *table_name* [:*partition_id*]] {in | out} *datafile*
[-c] [-E] [-n] [-N] [-X]
[-a *display_charset*]
[-A *packet_size*]
[-b *batchsize*]
[-e *errfile*]
[-f *formatfile*]
[-F *firstrow*]
[-g *id_start_value*]
[-I *interfaces_file*]
[-J *client_charset*]
[-L *lastrow*]
[-m *maxerrors*]
[-P *password*]
[-q *datafile_charset*]
[-r *row_terminator*]
[-R *remote_server_principal*]
[-S *server*]
[-t *field_terminator*]
[-T *text_or_image_size*]
[-U *username*]
[-z *language*]

or

bcp -v

Parameters

database_name
is optional if the table being copied is in your default database or in *master*. Otherwise, you must specify a database name.

owner
is optional if you or the Database Owner own the table being copied. If you do not specify an owner, **bcp** first looks for a table of that name that you own, and then looks for one owned by the Database Owner. If another user owns the table, you must specify the owner name or the command fails.

view_name
is the name of the view you are copying out.

table_name
is the name of the database table to copy. The table name cannot be a Transact-SQL reserved word.

Partition number *partition_number* does not exist in table *table_name*.

: *partition_id*
is the identifier of the partition into which to copy.

in | **out**
is the direction of the copy. **in** indicates a copy from a file into the database table; **out** indicates a copy to a file from the database table or view.

datafile
is the full path name of an operating system file. The path name can be from 1–255 characters in length.

-**c**
performs the copy operation with char datatype as the default storage type of all columns in the data file. Use this format if you are sharing data between platforms. This parameter does not prompt for each field; it uses char as the default storage type, no prefixes, \t (tab) as the default field terminator, and \n (newline) as the default row terminator.

-**E**
explicitly specifies values for a table's IDENTITY column.

By default, when you bulk copy data into a table with an IDENTITY column, **bcp** assigns each row a temporary IDENTITY column value of 0. This is effective only when copying data into a table. **bcp** reads the value of the ID column from the data file, but does not send it to the server. Instead, as **bcp** inserts each row into the table, the server assigns the row a unique, sequential, IDENTITY column value, beginning with the value 1. If you specify the **-E** flag when copying data into a table, **bcp** reads the value from the data file and sends it to the server which inserts the value into the table. If the number of rows inserted exceeds the maximum possible IDENTITY column value, Adaptive Server returns an error.

The -**E** parameter has no effect when you are bulk copying data out. Adaptive Server copies the ID column to the data file, unless you use the **-N** parameter.

You cannot use the -**E** and **-g** flags together.

-n

performs the copy operation using native (operating system) formats. Specifying the **-n** parameter means **bcp** will not prompt for each field. Files in native data format are not human-readable.

Warning! Do not use **bcp** in native format for data recovery or salvage or to resolve an emergency situation. Do not use **bcp** in native format to transport data between different hardware platforms, different operating systems, or different major releases of Adaptive Server. Do not use field terminators (**-t**) or row terminators (**-r**) with **bcp** in native format. Results are unpredictable and data could become corrupted. Using **bcp** in native format can create flat files that cannot be reloaded into Adaptive Server and it may be impossible to recover the data. If you are unable to re-run **bcp** in character format (for example, a table was truncated or dropped, hardware damage occurred, a database was dropped, and so on,) the data will be unrecoverable.

-N

skips the IDENTITY column. Use this parameter when copying data in if your host data file does not include a placeholder for the IDENTITY column values, or when copying data out, if you do not want to include the IDENTITY column information in the host file.

You cannot use both **-N** and **-E** parameters when copying data in.

-X

specifies that, in this connection to the server, the application initiates the login with client-side password encryption. **bcp** (the client) specifies to the server that password encryption is desired. The server sends back an encryption key, which **bcp** uses to encrypt your password. The server uses the key to authenticate your password when it arrives.

If **bcp** crashes, the system creates a core file that contains your password. If you did not use the encryption option, the password appears in plain text in the file. If you used the encryption option, your password is not readable.

-a *display_charset*

allows you to run **bcp** from a terminal where the character set differs from that of the machine on which **bcp** is running. Use **-a** in conjunction with **-J** to specify the character set translation file (*.xlt* file) required for the conversion. Use **-a** without **-J** only if the client character set is the same as the default character set.

The following error message will appear if the character translation file(s) named with the **-a** parameter is missing, or you mistype the name(s):

```
Error in attempting to determine the size of a pair of
translation tables.:'stat' utility failed.
```

-**A** *packet_size*
specifies the network packet size to use for this **bcp** session. For example:

```
bcp pubs2..titles out table_out -A 2048
```

sets the packet size to 2048 bytes for this **bcp** session. *size* must be between the values of the **default network packet size** and **maximum network packet size** configuration variables, and it must be a multiple of 512.

Use network packet sizes larger than the default to improve the performance of large bulk-copy operations.

-**b** *batchsize*
is the number of rows per batch of data copied (the default is to copy all the rows in one batch). Batching applies only when you are bulk copying in; it has no effect on bulk copying out.

Note Setting the batch size to 1 causes Adaptive Server to allocate one data page to each row copied in. This option only applies to fast bcp, and is only useful in locating corrupt rows of data.

-**e** *errfile*
is the full path name of an error file where **bcp** stores any rows that it was unable to transfer from the file to the database. Error messages from the **bcp** program appear on your terminal. **bcp** creates an error file only when you specify this parameter.

-**f** *format_file*
is the full path name of a file with stored responses from a previous use of **bcp** on the same table. After you answer **bcp**'s format questions, it asks you if you want to save your answers in a format file; creation of the format file is optional. The default file name is *bcp.fmt*. The **bcp** program can refer to a format file when copying data, so that you do not have to duplicate your previous format responses interactively. Use the **-f** parameter only if you previously created a format file that you want to use now for a copy in or out. If you do not specify this parameter, **bcp** queries you for format information interactively.

-**F** *firstrow*
is the number of the first row to copy from an input file (default is the first row).

-**g** *id_start_value*
specifies the value of the IDENTITY column to use as a starting point for copying data in.

You cannot use the -**g** and **-E** flags together.

-**I** *interfaces_file*
specifies the name and location of the interfaces file to search when connecting to Adaptive Server. If you do not specify **-I**, **bcp** looks for an interfaces file located in the directory specified by the SYBASE environment variable.

-**J** *client_charset*
specifies the character set to use on the client. **bcp** uses a filter to convert input between *client_charset* and the Adaptive Server character set.

-**J** *client_charset* requests that Adaptive Server convert to and from *client_charset*, the character set used on the client.

-**J** with no argument sets character-set conversion to NULL. No conversion takes place. Use this if the client and server use the same character set.

Omitting **-J** sets the character set to a default for the platform. Table 5-2 lists platform defaults.

Table 5-2: Default character sets for different platforms

Platform	Default Character Set
Sun Solaris, Digital UNIX, NCR, RS/6000	iso_1
HP-UX	roman8
OS/2, Novell NetWare 386	cp850
Macintosh	mac

The following error message will appear if an incorrect or unrecognized character set is named with the **-J** parameter:

```
Unrecognized localization object. Using default value
'iso_1'.
Starting copy...
=> warning.
```

For more information about character sets and associated flags, see the *System Administration Guide*.

-**L** *lastrow*
is the number of the last row to copy from an input file (default is the last row).

-**m** *maxerrors*
is the maximum number of nonfatal errors permitted before **bcp** aborts the copy. **bcp** discards each row that it cannot insert (due to a data conversion error, or an attempt to insert a null value into a column that does not allow them), counting each rejected row as one error. If you do not include this parameter, **bcp** uses a default value of 10.

-**P** *password*
specifies an Adaptive Server password. If you do not specify **-P** *password*, **bcp** prompts for a password.You can leave out the **-P** flag If your password is NULL.

-**q** *datafile_charset*
runs **bcp** to copy character data to or from a file system that uses a character set different from the client character set. Use **-q** in conjunction with **-J** to specify the character set translation file (*.xlt* file) required for the conversion.

In Japanese language environments, the **-q** flag translates Hankaku Katakana (half-width characters) into Zenkaku Katakana (full-width characters). Use with the argument *zenkaku* and with the **-J** flag to indicate the client's Japanese character set (sjis or eucjis). The *zenkaku.xlt* file translates *only* from the terminal to Adaptive Server, *not* from Adaptive Server to the terminal.

The following error message will appear if the character translation file(s) named with the **-q** parameter is missing, or you mistype the name(s):

```
Error in attempting to determine the size of a pair of
translation tables.:'stat' utility failed.
```

Note The ascii_7 character set is compatible with all character sets. If either the Adaptive Server character set or the client character set is set to ascii_7, any 7-bit ASCII character is allowed to pass between client and server unaltered. Other characters produce conversion errors. Character-set conversion issues are covered more thoroughly in the *System Administration Guide*.

-**r** *row_terminator*
– specifies the row terminator.

Warning! Do not use **-t** or **-r** parameters with **bcp** in native format. Results are unpredictable and data could become corrupted.

Note When specifying terminators from the command line with the **-t** or **-r** parameter, you must escape characters that have special significance to the UNIX operating system (see the examples in this section). Either place a backslash in front of the special character or enclose it in quotes. This is not necessary when **bcp** prompts you (interactive mode).

-**R** *remote_server_principal*
specifies the principal name for the remote server. By default, a server's principal name matches the server's network name (which is specified with the **-S** parameter or the DSQUERY environment variable). Use the **-R** parameter when the server's principal name and network name are not the same.

-**S** *server*
specifies the name of the Adaptive Server to which to connect. If you specify **-S** with no argument, **bcp** uses the server that your DSQUERY environment variable specifies.

-**t** *field_terminator*
specifies the field terminator.

Warning! Do not use **-t** or **-r** parameters with **bcp** in native format. Results are unpredictable and data could become corrupted.

-**T** *text_or_image_size*
allows you to specify, in bytes, the maximum length of text or image data that Adaptive Server sends. The default is 32K. If a text or image field is larger than the value of **-T** or the default, **bcp** does not send the overflow.

-**U** *username*
specifies an Adaptive Server login name. If you do not specify *username*, **bcp** uses the current user's operating system login name.

-**z** *language*
is the official name of an alternate language that the server uses for date formats and to display **bcp** prompts and messages. Without the **-z** flag, **bcp** uses the server's default language. You can add languages to an Adaptive Server during installation or afterwards, using either the **langinstall** utility or the **sp_addlanguage** stored procedure.

The following error message will appear if an incorrect or unrecognized language is named with the **-z** parameter:

```
Unrecognized localization object. Using default value
'us_english'.
Starting copy...
=> warning.
```

-**v**
displays the version number of **bcp** and a copyright message and returns to the operating system.

Examples

```
bcp pubs2..publishers out pub_out -c -t , -r \\r
```

Copies data out of the *publishers* table. The **-c** parameter specifies character format (using char for all fields). The **-t** *field_terminator* parameter ends each field with a comma, and the **-r** *row_terminator* parameter ends each line with a Return. **bcp** prompts only for a password. The first backslash before the final "r" escapes the second so that only one backslash is printed.

```
bcp pubs2..publishers out pub_out

Password:
Enter the file storage type of field pub_id [char]:
Enter prefix length of field pub_id [0]:
Enter length of field pub_id [4]:
Enter field terminator [none]:
Enter the file storage type of field pub_name [char]:
Enter prefix length of field pub_name [1]:
Enter length of field pub_name [40]:
Enter field terminator [none]:
Enter the file storage type of field city [char]:
Enter prefix length of field city [1]:
Enter length of field city [20]:
Enter field terminator [none]:
Enter the file storage type of field state [char]:
Enter prefix length of field state [1]:
Enter length of field state [2]:
Enter field terminator [none]:
Do you want to save this format information in a
 file? [Y-n] y
```

```
Host filename [bcp.fmt]: pub_formStarting copy...
3 rows copied.
 Clock Time (ms.): total = 1 Avg = 0 (3000.00 rows per
sec.)
```

Copies data from the *publishers* table to a file named *pub_out* for reloading later into Adaptive Server. Press Return to accept the defaults specified by the prompts. The same prompts appear when you copy data into the *publishers* table.

```
bcp pubs2..publishers in pub_out -f pub_form
```

Copies data back into Adaptive Server using the saved format file, *pub_form*.

```
To see examples of datatypes, enter "?" at the prompt:

Enter the file storage type of field 'pub_id'
 ['char']:?
Invalid column type. Valid types are:
 <cr>: same type as Adaptive Server column.
 c : char
 T : text
 i : int
 s : smallint
 t : tinyint
 f : float
 m : money
 b : bit
 d : datetime
 x : binary
 I : image
 D : smalldatetime
 r : real
 M : smallmoney
 n : numeric
 e : decimal
```

Enter the single letter exactly as it appears above.

```
bcp pubs2..publishers in vt200_data -J iso_1 -q vt200 -
z french
```

Copies a data file created with a character set used on a VT200 terminal into the *pubs2..publishers* table. The **-q** flag translates it. The **-z** flag displays **bcp** messages in French.

```
bcp pubs2..publishers in -a mac -J roman8 -q iso_1
```

Specifies that you are using a Macintosh, running **bcp** on a workstation that is using roman8, with the file system on another machine that uses iso_1.

```
bcp pubs2..publishers out -T 40960 -A 4096
```

Specifies that Adaptive Server send 40K of text or image data using a packet size of 4096 bytes.

Usage

- See Chapter 2, "Using bcp to Transfer Data to and from Adaptive Server" for an in-depth discussion of **bcp**.
- See the *Performance and Tuning Guide* for more information on how changing certain parameters can affect **bcp** for large batches.
- The current version of **bcp** ignores the **-y** *sybase_directory* parameter.
- **bcp** provides a convenient, high-speed method for transferring data between a database table or view and an operating system file. **bcp** can read or write files in a wide variety of formats. When copying in from a file, **bcp** appends data to an existing database table; when copying out to a file, **bcp** overwrites any previous contents of the file.
- You cannot copy into a view from an operating system file.
- Upon completion, **bcp** informs you of the number of rows of data successfully copied, the number of rows (if any) that it could not copy, the total time the copy took, the average amount of time, in milliseconds, that it took to copy one row and the number of rows copied per second.
- **bcp** copies each batch in a single transaction. If Adaptive Server rejects any row in the batch, the entire transaction is rolled back. By default, **bcp** copies all rows in a single batch; use the **-b** parameter to change the default batch size. Adaptive Server considers each batch a single **bcp** operation, writes each batch to a separate data page, and continues to the next batch, regardless of whether the previous transaction succeeded.
- When copying data into a table, **bcp** observes any defaults defined for columns and user-defined datatypes. However, **bcp** ignores rules and triggers in order to load data at the fastest possible speed.
- When you send host data files to sites using terminals that are different from your own, inform them of the *datafile_charset* that you used to create the files.
- Do not use **-t** or **-r** parameters with **bcp** in native format. Results are unpredictable and data could become corrupted.

Fast and Slow *bcp*

- The **bcp** program is optimized to load data into tables that do not have indexes or triggers associated with them. It loads data into tables without indexes or triggers at the fastest possible speed, with a minimum of logging. Page allocations are logged, but the insertion of rows is not.

  When you copy data into a table that has one or more indexes or triggers, a slower version of **bcp** is automatically used, which logs row inserts. This includes indexes implicitly created using the unique integrity constraint of a **create table** statement. However, **bcp** does not enforce the other integrity constraints defined for a table.

- Because changes to data are not logged when using fast **bcp**, you must use **dump database** to back up your databases. You cannot use **dump transaction** after a fast bulk copy because no transactions are added to the transaction log.

  Because the fast version of **bcp** inserts data without logging it, the System Administrator or Database Owner must first set the system procedure **sp_dboption**, **"DB"**, to **true**. If the option is not set to **true**, and you try to copy data into a table that has no indexes or triggers, Adaptive Server generates an error message. You do not need to set this option to copy data out to a file or to copy data into a table that contains indexes or triggers.

  By default, the **select into/bulkcopy/pllsort** option is **false** (off) in newly created databases. To change the default situation, turn this option on in the *model* database.

- Fast **bcp** runs more slowly while a **dump database** is taking place.

  The log can grow very large during slow **bcp**. After the bulk copy completes, back up your database with **dump database**, then truncate the log with **dump transaction**. You may want to use **trunc log on chkpt** or a threshold procedure to perform log dumps.

Note The performance penalty for copying data into a table that has indexes or triggers in place can be severe. If you are copying in a very large number of rows, it may be faster to drop all the indexes and triggers first with **drop index** (or **alter table** for indexes created as a unique constraint) and **drop trigger**, set the database option, copy the data into the table, re-create the indexes and triggers, and then dump the database. Remember to allocate disk space for the construction of indexes and triggers—for a clustered index, about 1.2 times the amount of space needed for the data, in addition to the space needed for the data.

Responding to *bcp* Prompts

When you copy data in or out using the - **n** (native format) or **-c** (character format) parameter, **bcp** prompts you only for your password, unless you supplied it with the **-P** parameter. If you do not supply either the **-n** or **-c** parameter, **bcp** prompts you for information for each field (column) in the table or view.

- Each prompt displays a default value, in brackets, which you can accept by pressing Return. The prompts include:
  - The file storage type, which can be *char* or any valid Adaptive Server datatype
  - The prefix length, which is an integer indicating the length in bytes of the following data
  - The storage length of the data in the file
  - The field terminator, which can be any character string

  The row terminator is the field terminator of the last field in the table or file.
- The bracketed defaults represent reasonable values for the datatypes of the field in question. For the most efficient use of space when copying out to a file:
  - Use the default prompts
  - Copy all data in the datatypes defined by their table
  - Use prefixes as indicated
  - Do not use terminators
  - Accept the default lengths

  Table 5-3 shows the **bcp** prompts, default user responses, and possible alternate responses:

Table 5-3: bcp prompts—their defaults and user responses

Prompt	Default Provided	Possible User Response
File Storage Type	Use database storage type for most fields except: *char* for varchar binary for varbinary	char, to create or read a human-readable file; any Adaptive Server datatype where implicit conversion is supported.
Prefix Length	0 for fields defined with char datatype (not storage type) and all fixed-length datatypes 1 for most other datatypes 2 for binary and varbinary saved as char 4 for text and image	0 if no prefix is desired; otherwise, defaults are recommended.
Storage Length	For char and varchar, use defined length. For binary and varbinary saved as char, use double the defined length. For all other datatypes, use maximum length needed to avoid truncation or data overflow.	Default values, or greater, are recommended.
Field or Row Terminator	None	Up to 30 characters

Note Be careful when you copy data from different operating systems because not all operating systems use the same native datatypes (for example, copying from NT into a UNIX server).

- A prefix length is a 1-, 2-, or 4-byte integer that represents the length of each data value in bytes. It immediately precedes the data value in the host file.
- A field terminator string can be up to 30 characters long. The most common terminators are a tab (entered as "\t" and used for all columns except the last one), and a newline (entered as "\n" and used for the last field in a row). Other terminators are:"\0" (the null terminator), "\" (backslash), and "\r" (Return). When choosing a terminator, be sure that its pattern does not appear in any of your character data, because **bcp** always looks for the first possible terminator.

  A terminator or prefix affects the actual length of data transferred:

- If the length of an entry being copied out to a file is less than the storage length (other than char, nchar, and binary data), it is followed immediately by the terminator or the prefix for the next field, and not padded to the full storage length.
- When copying in, **bcp** copies data until it either reaches the number of bytes indicated in the "Length" prompt or encounters the terminator. If the data exceeds the specified length, **bcp** flushes the data until it finds the terminator. Without terminators, the table storage length is strictly observed.

- Fields stored as char (except char, nchar, and binary fields) instead of their database datatypes take less file storage space with the default length and prefix or a terminator. **bcp** can use either a terminator or a prefix to determine the most efficient use of storage space. **bcp** suggests the maximum amount of storage space required for each field as the default. For char or varchar data, **bcp** accepts any length.
- Fields defined in the database as char, nchar, and binary, and those that do not permit null values, are always padded with spaces (null bytes for binary) to the full length defined in the database. timestamp data is treated as binary(8).

  If data in the varchar and varbinary fields is longer than the length specified for copy out, **bcp** silently truncates the data in the file at the specified length.
- Table 5-4 and Table 5-5 show the interaction of prefix lengths, terminators, and field lengths on the information in the file. "P" indicates the prefix in the stored table. "T" indicates the terminator, and dashes (--) show appended spaces. An ellipsis (...) indicates that the pattern repeats for each field. The field length is 8 for each column, and "string" represents the 6-character field each time.

Table 5-4: Adaptive Server char data

	Prefix length = 0	***Prefix length = 1, 2, or 4***
No terminator	string--string--.	Pstring--Pstring--.
Terminator	string--Tstring--T.	Pstring--TPstring--T.

Table 5-5: Other datatypes converted to char storage

	Prefix length = 0	***Prefix length = 1, 2, or 4***
No terminator	string--string--.	PstringPstring.
Terminator	stringTstringT.	PstringTPstringT.

- File storage length is the maximum amount of data that can be transferred for the column, plus terminators and prefixes.
- **bcp** considers any data column that can contain null values to be variable length, so use either a length prefix or terminator to denote the length of each row of data.

Datatypes and Storage Types

- **bcp** can copy data out to a file either as its database datatype or as any datatype for which implicit conversion is supported for the datatype in question. **bcp** copies user-defined datatypes as their base datatype or as any datatype for which implicit conversion is supported. For more information on datatype conversions, see **dbconvert** in the *Open Client DB-Library/C Reference Manual* and "Datatype Conversion Functions" in the *Adaptive Server Reference Manual*.
- The file storage type and length of a column need not be the same as the type and length of the column in the database table. (If types and formats copied in are incompatible with the structure of the database table, the copy fails.)
- Data written to a host file in its native format preserves all of its precision. datetime and float values preserve all of their precision even when they are converted to character format. Adaptive Server stores money values to a precision of one ten-thousandth of a monetary unit. However, when money values are converted to character format, their character format values are recorded only to the nearest two places.
- Before copying data that is in character format from a file into a database table, check the datatype entry rules in the *Adaptive Server Reference Manual*. Character data copied into the database with **bcp** must conform to those rules. Note especially that dates in the undelimited *(yy)yymmdd* format may result in overflow errors if the year is not specified first.

See also

System procedures	sp_audit, sp_dboption, sp_displayaudit
System tables	sysaudits_01 – sysaudits_08

bcp_dce

Description

Copies a database table to or from an operating system file in a user-specified format.

Syntax

```
bcp_dce [[database_name.]owner.][view_name | table_name
        [:partition_id]] {in | out} datafile
[-c] [-E] [-n] [-N] [-v] [-X]
 [-a display_charset]
 [-A packet_size]
 [-b batchsize]
 [-e errfile]
 [-f formatfile]
 [-F firstrow]
 [-g id_start_value]
 [-I interfaces_file]
 [-J client_charset]
 [-K keytab_file]
 [-L lastrow]
 [-m maxerrors]
 [-M LabelName LabelValue] [-labeled]
 [-P password]
 [-q datafile_charset]
 [-r row_terminator]
 [-R remote_server_principal]
 [-S server]
 [-t field_terminator]
 [-T text_or_image_size]
 [-U username]
 [-V [security_options]
 [-y sybase_dir]
 [-z language]
 [-Z security_mechanism]
```

Parameters

database_name

is optional if the table being copied is in your default database or in *master*. Otherwise, you must specify a database name.

owner

is optional if you or the Database Owner own the table being copied. If you do not specify an owner, **bcp_dce** first looks for a table of that name that you own, and then looks for one owned by the Database Owner. If another user owns the table, you must specify the owner name or the command fails.

table_name

is the name of the database table to copy. The table name cannot be a Transact-SQL reserved word.

Partition number *partition_number* does not exist in table *table_name*.

view_name
is the name of the view you are copying out.

in | **out**
is the direction of the copy. **in** indicates a copy from a file into the database table; **out** indicates a copy to a file from the database table or view.

datafile
is the full path name of an operating system file. The path name can be from 1–255 characters in length.

-**c**
performs the copy operation with char datatype as the default storage type of all columns in the data file. Use this format if you are sharing data between platforms. This parameter does not prompt for each field; it uses char as the default storage type, no prefixes, \t (tab) as the default field terminator, and \n (newline) as the default row terminator.

-**E**
explicitly specifies values for a table's IDENTITY column.

By default, when you bulk copy data into a table with an IDENTITY column, **bcp_dce** assigns each row a temporary IDENTITY column value of 0. This is effective only when copying data into a table. **bcp_dce** reads the value of the ID column from the data file, but does not send it to the server. Instead, as **bcp_dce** inserts each row into the table, the server assigns the row a unique, sequential, IDENTITY column value, beginning with the value 1. If you specify the **-E** flag when copying data into a table, **bcp_dce** reads the value from the data file and sends it to the server which inserts the value into the table. If the number of rows inserted exceeds the maximum possible IDENTITY column value, Adaptive Server returns an error.

The -**E** parameter has no effect when you are bulk copying data out. Adaptive Server copies the ID column to the data file, unless you use the **-N** parameter.

You cannot use the -**E** and **-g** flags together.

-**labeled**
(secure Adaptive Server only) for importing data this parameter indicates that the data you are importing already has labels in the first field of every record.

For exporting data this parameter indicates that you want the sensitivity label of every row to be copied out as the first field.

-M *label_name* "*label_value*"
(secure SQL server only) enables multilevel users to set the session labels for the bulk-copy. Valid values for *label_name* are:

- *curread* (current read level) is the initial level of data that you can read during this session. *curread* must dominate *curwrite*.
- *curwrite* (current write level) is the initial sensitivity level that will be applied to any data that you write during this session.
- *maxread* (maximum read level) is the maximum level at which you can read data. This is the upper bound to which you as a multilevel user can set your *curread* during the session. *maxread* must dominate *maxwrite*.
- *maxwrite* (maximum write level) is the maximum level at which you can write data. This is the upper bound to which you as a multilevel user can set your *curwrite* during a session. *maxwrite* must dominate *minwrite* and *curwrite*.
- *minwrite* (minimum write level) is the minimum level at which you can write data. This is the lower bound to which you as a multilevel user can set *curwrite* during a session. *minwrite* must be dominated by *maxwrite* and *curwrite*.

label_value is the actual value of the label, expressed in the human-readable format used on your system (for example, "Company Confidential Personnel").

-n
performs the copy operation using native (operating system) formats. This parameter does not prompt for each field. Files in native data format are not human-readable.

Warning! Do not use **-t** or **-r** parameters with **bcp_dce** in native format. Results are unpredictable and data could become corrupted.

-N
skips the IDENTITY column. Use this parameter when copying data in if your host data file does not include a placeholder for the IDENTITY column values, or when copying data out, if you do not want to include the IDENTITY column information in the host file.

You cannot use both **-N** and **-E** parameters when copying data in.

-v
displays the version number of **bcp_dce** and a copyright message and returns to the operating system.

-X
When connecting to the server, **bcp_dce** initiates the login with client-side password encryption. **bcp_dce** (the client) specifies to the server that password encryption is desired. The server sends back an encryption key, which **bcp_dce** uses to encrypt your password. The server uses the key to authenticate your password when it arrives.

If **bcp_dce** crashes, the system creates a core file that contains your password. If you did not use the encryption option, the password appears in plain text in the file. If you used the encryption option, your password is not readable.

-a *display_charset*
allows you to run **bcp_dce** from a terminal where the character set differs from that of the machine on which **bcp_dce** is running. Use **-a** in conjunction with **-J** to specify the character set translation file (*.xlt* file) required for the conversion. Use **-a** without **-J** only if the client character set is the same as the default character set.

The following error message will appear if the character translation file(s) named with the **-a** parameter is missing, or you mistype the name(s):

```
Error in attempting to determine the size of a pair of
translation tables.:'stat' utility failed.
```

-A *packet_size*
specifies the network packet size to use for this **bcp_dce** session. For example:

```
bcp_dce pubs2..titles out table_out -A 2048
```

sets the packet size to 2048 bytes for this **bcp_dce** session. *size* must be between the values of the **default network packet size** and **maximum network packet size** configuration variables, and it must be a multiple of 512.

Larger network packet sizes improve the performance of large bulk-copy operations.

-**b** *batchsize*
is the number of rows per batch of data copied (the default is to copy all the rows in one batch). Batching applies only when you are bulk copying in; it has no effect on bulk copying out.

Note Setting the batch size to 1 causes Adaptive Server to allocate one data page to each row copied in. This option only applies to fast bcp_dce, and is only useful in locating corrupt rows of data.

-**e** *errfile*
is the full path name of an error file where **bcp_dce** stores any rows that it was unable to transfer from the file to the database. Error messages from the **bcp_dce** program appear on your terminal. **bcp_dce** creates an error file only when you specify this parameter.

-**f** *format_file*
is the full path name of a file with stored responses from a previous use of **bcp_dce** on the same table. After you answer **bcp_dce**'s format questions, it asks you if you want to save your answers in a format file; creation of the format file is optional. The default file name is *bcp_dce.fmt*. The **bcp_dce** program can refer to a format file when copying data, so that you do not have to duplicate your previous format responses interactively. Use this parameter only if you previously created a format file that you want to use now for a copy in or out. If this parameter is not used, **bcp_dce** queries for format information interactively.

-**F** *firstrow*
is the number of the first row to copy from an input file (default is the first row).

-**g** *id_start_value*
specifies the value of the IDENTITY column to use as a starting point for copying data in.

You cannot use the -**g** and **-E** flags together.

-**I** *interfaces_file*
specifies the name and location of the interfaces file to search when connecting to Adaptive Server. If you do not specify **-I**, **bcp_dce** looks for an interfaces file located in the directory specified by the SYBASE environment variable.

-**J** *client_charset*
specifies the character set to use on the client. **bcp_dce** uses a filter to convert input between *client_charset* and the Adaptive Server character set.

-**J** *client_charset* requests that Adaptive Server convert to and from *client_charset*, the character set used on the client.

-**J** with no argument sets character-set conversion to NULL. No conversion takes place. Use this if the client and server use the same character set.

Omitting **-J** sets the character set to a default for the platform. Table 5-2 lists platform defaults.

Table 5-6: Default character sets for different platforms

Platform	Default Character Set
Sun Solaris, Digital UNIX, NCR, RS/6000	iso_1
HP-UX	roman8
OS/2, Novell NetWare 386	cp850
Macintosh	mac

The following error message will appear if an incorrect or unrecognized character set is named with the **-J** parameter:

```
Unrecognized localization object. Using default value
'iso_1'.
Starting copy...
=> warning.
```

-**L** *lastrow*
is the number of the last row to copy from an input file (default is the last row).

-**m** *maxerrors*
is the maximum number of nonfatal errors permitted before **bcp_dce** aborts the copy. **bcp_dce** discards each row that it cannot insert (due to a data conversion error, or an attempt to insert a null value into a column that does not allow them), counting each rejected row as one error. If you do not include this parameter, **bcp_dce** uses a default value of 10.

-**P** *password*
specifies an Adaptive Server password. If you do not specify **-P** *password*, **bcp_dce** prompts for a password. You can leave out the **-P** flag if your password is NULL.

Note The utility programs described in Table 5-2 may allow you to use a **-P** parameter to enter your password. If security is an issue, do not use this parameter to specify your password. Another user may have an opportunity to see it. Instead, log in as usual without the **-P** parameter, and let Adaptive Server prompt you for your password.

-**q** *datafile_charset*
allows you to run **bcp_dce** to copy character data to or from a file system that uses a character set different from the client character set. **-q** in conjunction with **-J** specifies the character set translation file (*.xlt* file) required for the conversion.

In Japanese language environments, the **-q** flag translates Hankaku Katakana (half-width characters) into Zenkaku Katakana (full-width characters). Use with the argument *zenkaku* and with the **-J** flag to indicate the client's Japanese character set (sjis or eucjis). The *zenkaku.xlt* file translates *only* from the terminal to Adaptive Server, *not* from Adaptive Server to the terminal.

The following error message will appear if the character translation file(s) named with the **-q** parameter is missing, or you mistype the name(s):

```
Error in attempting to determine the size of a pair of
translation tables.:'stat' utility failed.
```

Note The ascii_7 character set is compatible with all character sets. If either the Adaptive Server character set or the client character set is set to ascii_7, any 7-bit ASCII character is allowed to pass between client and server unaltered. Other characters produce conversion errors. Character-set conversion issues are covered more thoroughly in the *System Administration Guide*.

-**r** *row_terminator*
specifies the row terminator.

Warning! Do not use **-t** or **-r** parameters with **bcp_dce** in native format. Results are unpredictable and data could become corrupted.

Note When specifying terminators from the command line with the **-t** or **-r** parameter, you must escape characters that have special significance to the UNIX operating system (see the examples in this section). Either place a backslash in front of the special character or enclose it in quotes. This is not necessary when **bcp_dce** prompts you (interactive mode).

-**R** *remote_server_principal*
specifies the principal name for the remote server. By default, a server's principal name matches the server's network name (which is specified with the **-S** parameter or the DSQUERY environment variable). Use the **-R** parameter when the server's principal name and network name are not the same.

-**S** *server*
specifies the name of the Adaptive Server to which to connect. If you specify **-S** with no argument, **bcp_dce** uses the server that your DSQUERY environment variable specifies.

-**t** *field_terminator*
specifies the field terminator.

Warning! Do not use **-t** or **-r** parameters with **bcp_dce** in native format. Results are unpredictable and data could become corrupted.

-**T** *text_or_image_size*
allows you to specify, in bytes, the maximum length of text or image data that Adaptive Server sends. The default is 32K. If a text or image field is larger than the value of **-T** or the default, **bcp_dce** does not send the overflow.

-**U** *username*
specifies an Adaptive Server login name. If you do not specify *username*, **bcp_dce** uses the current user's operating system login name.

-z *language*
is the official name of an alternate language that the server uses for date formats and to display **bcp_dce** prompts and messages. Without the **-z** flag, **bcp_dce** uses the server's default language. You can add languages to an Adaptive Server during installation or afterwards, using either the **langinstall** utility or the **sp_addlanguage** stored procedure.

The following error message will appear if an incorrect or unrecognized language is named with the **-z** parameter:

```
Unrecognized localization object. Using default value
'us_english'.
Starting copy...
=> warning.
```

Examples

```
bcp_dce pubs2..publishers out pub_out -c -t , -r \\r
```

Copies data out of the *publishers* table. The **-c** parameter specifies character format (using char for all fields). The **-t** *field_terminator* parameter ends each field with a comma, and the **-r** *row_terminator* parameter ends each line with a Return. **bcp_dce_dce** prompts only for a password. The first backslash before the final "r" escapes the second so that only one backslash is printed.

```
bcp_dce_dce pubs2..publishers out pub_out

Password:
Enter the file storage type of field pub_id [char]:
Enter prefix length of field pub_id [0]:
Enter length of field pub_id [4]:
Enter field terminator [none]:
Enter the file storage type of field pub_name [char]:
Enter prefix length of field pub_name [1]:
Enter length of field pub_name [40]:
Enter field terminator [none]:
Enter the file storage type of field city [char]:
Enter prefix length of field city [1]:
Enter length of field city [20]:
Enter field terminator [none]:
Enter the file storage type of field state [char]:
Enter prefix length of field state [1]:
Enter length of field state [2]:
Enter field terminator [none]:
Do you want to save this format information in a
 file? [Y-n] y
Host filename [bcp_dce.fmt]: pub_form
Starting copy...
3 rows copied.
Clock Time (ms.): total = 1 Avg = 0 (3000.00 rows per
```

```
sec.)
```

Copies data from the *publishers* table to a file named *pub_out* for reloading later into Adaptive Server. Press Return to accept the defaults specified by the prompts. The same prompts appear when you copy data into the *publishers* table.

```
bcp_dce pubs2..publishers in pub_out -f pub_form
```

Copies data back into Adaptive Server using the saved format file, *pub_form*.

```
To see examples of datatypes, enter "?" at the prompt:

Enter the file storage type of field 'pub_id'
 ['char']:?
Invalid column type. Valid types are:
 <cr>: same type as Adaptive Server column.
 c : char
 T : text
 i : int
 s : smallint
 t : tinyint
 f : float
 m : money
 b : bit
 d : datetime
 x : binary
 I : image
 D : smalldatetime
 r : real
 M : smallmoney
 n : numeric
 e : decimal
```

Enter the single letter exactly as it appears above.

```
bcp_dce pubs2..publishers in vt200_data -J iso_1 -q
vt200 -z french
```

Copies a data file created with a character set used on a VT200 terminal into the *pubs2..publishers* table. The **-q** flag translates it. The **-z** flag displays **bcp_dce** messages in French.

```
bcp_dce pubs2..publishers in -a mac -J roman8 -q iso_1
```

Specifies that you are using a Macintosh, running **bcp_dce** on a workstation that is using roman8, with the file system on another machine that uses iso_1.

```
bcp_dce pubs2..publishers out -T 40960 -A 4096
```

Specifies that Adaptive Server send 40K of text or image data using a packet size of 4096 bytes.

Usage

- **bcp_dce** is available only for the IBM RS/6000 platform of UNIX.
- See Chapter 2, "Using bcp to Transfer Data to and from Adaptive Server" for a more in-depth discussion of **bcp_dce**.
- See the *Performance and Tuning Guide* for more information on how changing certain parameters can affect **bcp_dce** for large batches.
- If you are running on a trusted operating system that supports mandatory access controls (MAC), you must log into the operating system at the same sensitivity label as Adaptive Server. See your Adaptive Server System Security Officer if you do not know Adaptive Server's sensitivity label.
- **bcp_dce** provides a convenient, high-speed method for transferring data between a database table or view and an operating system file. **bcp_dce** can read or write files in a wide variety of formats. When copying in from a file, **bcp_dce** appends data to an existing database table; when copying out to a file, **bcp_dce** overwrites any previous contents of the file.
- Upon completion, **bcp_dce** informs you of the number of rows of data successfully copied, the number of rows (if any) that it could not copy, the total time the copy took, the average amount of time that it took to copy one row (in milliseconds) and the number of rows copied per second.
- fast **bcp_dce** copies each batch in a single **insert** transaction. If Adaptive Server rejects any row in the batch, the entire **insert** is rolled back. By default, fast **bcp_dce** copies all rows in a single batch; use the **-b** parameter to change the default batch size. Adaptive Server considers each batch a single **bcp_dce** operation, writes each batch to a separate data page, and continues to the next batch, regardless of whether the **insert** succeeded.

  Note Setting the batch size to 1 causes Adaptive Server to allocate one data page to each row copied in. This option only applies to fast **bcp_dce**, and is only useful in locating corrupt rows of data.

- When copying data into a table, **bcp_dce** observes any defaults defined for columns and user-defined datatypes. However, **bcp_dce** ignores rules in order to load data at the fastest possible speed.
- When you send host data files to sites using terminals that are different from your own, inform them of the *datafile_charset* that you used to create the files.

- You cannot use named pipes to copy files in or out.
- Do not use **-t** or **-r** parameters with **bcp_dce** in native format. Results are unpredictable and data could become corrupted.

Fast and Slow *bcp_dce*

- The **bcp_dce** program is optimized to load data into tables that do not have indexes or triggers associated with them. It loads data into tables without indexes or triggers at the fastest possible speed, with a minimum of logging. Page allocations are logged, but the insertion of rows is not.

  When you copy data into a table that has one or more indexes or triggers, a slower version of **bcp_dce** is automatically used, which logs row inserts. This includes indexes implicitly created using the unique integrity constraint of a **create table** statement. However, **bcp_dce** does not enforce the other integrity constraints defined for a table.

- Because changes to data are not logged when using fast **bcp_dce**, you must use **dump database** to back up your databases. You cannot use **dump transaction** after a fast bulk copy because no transactions are added to the transaction log.

  Because the fast version of **bcp_dce** inserts data without logging it, the System Administrator or Database Owner must first set the system procedure **sp_dboption, "DB", true**. If the option is not **true**, and you try to copy data into a table that has no indexes or triggers, Adaptive Server generates an error message. You do not need to set this option to copy data out to a file or to copy data into a table that contains indexes or triggers.

  By default, the **select into/bulkcopy/pllsort** option is **false** (off) in newly created databases. To change the default situation, turn this option on in the *model* database.

- Fast **bcp_dce** runs more slowly while a **dump database** is taking place.

- The log can grow very large during slow **bcp_dce**. After the bulk copy completes, back up your database with **dump database**, then truncate the log with **dump transaction**. You may want to use **trunc log on chkpt** or a threshold procedure to perform log dumps.

Note The performance penalty for copying data into a table that has indexes or triggers in place can be severe. If you are copying in a very large number of rows, it may be faster to drop all the indexes and triggers first with **drop index** (or **alter table** for indexes created as a unique constraint) and **drop trigger**, set the database option, copy the data into the table, re-create the indexes and triggers, and then dump the database. Remember to allocate disk space for the construction of indexes and triggers—for a clustered index, about 1.2 times the amount of space needed for the data, in addition to the space needed for the data.

Responding to *bcp_dce* Prompts

When you copy data in or out using the **-n** (native format) or **-c** (character format) parameter, **bcp_dce** prompts you only for your password, unless you supplied it with the **-P** parameter. If you do not supply either the **-n** or **-c** parameters, **bcp_dce** prompts you for information for each field (column) in the table or view.

- Each prompt displays a default value, in brackets, which you can accept by pressing Return. The prompts include:
  - The file storage type, which can be *char* or any valid Adaptive Server datatype
  - The prefix length, which is an integer indicating the length in bytes of the following data
  - The storage length of the data in the file
  - The field terminator, which can be any character string

  The row terminator is the field terminator of the last field in the table or file.
- The bracketed defaults represent reasonable values for the datatypes of the field in question. For the most efficient use of space when copying out to a file:
  - Use the default prompts
  - Copy all data in their table datatypes

- Use prefixes as indicated
- Do not use terminators
- Accept the default lengths

Table 5-3 shows the **bcp_dce** prompts, default user responses, and possible alternate responses:

Table 5-7: bcp_dce prompts—their defaults and user responses

Prompt	Default Provided	Possible User Response
File Storage Type	Use database storage type for most fields except: *char* for varchar binary for varbinary	char, to create or read a human-readable file; any Adaptive Server datatype where implicit conversion is supported.
Prefix Length	0 for fields defined with char datatype (not storage type) and all fixed-length datatypes 1 for most other datatypes 2 for binary and varbinary saved as char 4 for text and image	0 if no prefix is desired; otherwise, defaults are recommended.
Storage Length	For char and varchar, use defined length. For binary and varbinary saved as char, use double the defined length. For all other datatypes, use maximum length needed to avoid truncation or data overflow.	Default values, or greater, are recommended.
Field or Row Terminator	None	Up to 30 characters

Note Be careful when you copy data from different operating systems because not all operating systems use the same native datatypes (for example, copying from NT into a UNIX server).

- A prefix length is a 1-, 2-, or 4-byte integer that represents the length of each data value in bytes. It immediately precedes the data value in the host file.

- A field terminator string can be up to 30 characters long. The most common terminators are a tab (entered as "\t" and used for all columns except the last one), and a newline (entered as "\n" and used for the last field in a row). Other terminators are: "\0" (the null terminator), "\" (backslash), and "\r" (Return). When choosing a terminator, be sure that its pattern does not appear in any of your character data, because **bcp_dce** always looks for the first possible terminator.

  A terminator or prefix affects the actual length of data transferred:

  - If the length of an entry being copied out to a file is less than the storage length (other than char, nchar, and binary data), it is followed immediately by the terminator or the prefix for the next field, and not padded to the full storage length.
  - When copying in, **bcp_dce** copies data until it either reaches the number of bytes indicated in the "Length" prompt or encounters the terminator. If the data exceeds the specified length, **bcp_dce** flushes the data until it finds the terminator. Without terminators, the table storage length is strictly observed.

- Fields stored as char (except char, nchar, and binary fields) instead of their database datatypes take less file storage space with the default length and prefix or a terminator. **bcp_dce** can use either a terminator or a prefix to determine the most efficient use of storage space. **bcp_dce** suggests the maximum amount of storage space required for each field as the default. For char or varchar data, **bcp_dce** accepts any length.

- Fields defined in the database as char, nchar, and binary, and those that do not permit null values, are always padded with spaces (null bytes for binary) to the full length defined in the database. timestamp data is treated as binary(8).

  If data in the varchar and varbinary fields is longer than the length specified for copy out, **bcp_dce** silently truncates the data in the file at the specified length.

- Table 5-4 and Table 5-5 show the interaction of prefix lengths, terminators, and field lengths on the information in the file. "P" indicates the prefix in the stored table. "T" indicates the terminator, and dashes (--) show appended spaces. An ellipsis (...) indicates that the pattern repeats for each field. The field length is 8 for each column, and "string" represents the 6-character field each time.

Table 5-8: Adaptive Server char data

	Prefix length = 0	***Prefix length = 1, 2, or 4***
No terminator	string--string--.	Pstring--Pstring--.
Terminator	string--Tstring--T.	Pstring--TPstring--T.

Table 5-9: Other datatypes converted to char storage

	Prefix length = 0	***Prefix length = 1, 2, or 4***
No terminator	string--string--.	PstringPstring.
Terminator	stringTstringT.	PstringTPstringT.

- File storage length is the maximum amount of data that can be transferred for the column, plus terminators and prefixes.
- **bcp_dce** considers any data column that can contain null values to be variable length, so use either a length prefix or terminator to denote the length of each row of data.

Datatypes and Storage Types

- **bcp_dce** can copy data out to a file either as its database datatype or as any datatype for which implicit conversion is supported for the datatype in question. **bcp_dce** copies user-defined datatypes as their base datatype or as any datatype for which implicit conversion is supported. For more information on datatype conversions, see **dbconvert** in the *Open Client DB-Library/C Reference Manual* and "Datatype Conversion Functions" in the *Adaptive Server Reference Manual*.
- The file storage type and length of a column need not be the same as the type and length of the column in the database table. (If types and formats copied in are incompatible with the structure of the database table, the copy fails.
- Data written to a host file in its native format preserves all of its precision. datetime and float values preserve all of their precision even when they are converted to character format. Adaptive Server stores money values to a precision of one ten-thousandth of a monetary unit. However, when money values are converted to character format, their character format values are recorded only to the nearest two places.
- Before copying data that is in character format from a file into a database table, check the datatype entry rules in the "Datatypes" section of the *Adaptive Server Reference Manual*. Character data that is being copied into the database with **bcp_dce** must conform to those rules. Note especially that dates in the undelimited *(yy)yymmdd* format may result in overflow errors if the year is not specified first.

See also

System procedures	sp_audit, sp_dboption, sp_displayaudit
System tables	sysaudits_01 – sysaudits_08

buildmaster

Description

Builds the master device and creates the *master*, *model*, and *tempdb* databases on the device.

Syntax

```
buildmaster
 [-d disk]
 [-c cno]
 [-s size]
 [-m]
 [-r]
 [-q]
 [-x]
```

or

```
buildmaster -v
```

Parameters

-**d** *disk*
is the physical name of the raw disk partition or operating system file where the master device resides.

-**c** *cno*
is the controller number for the master device. Together, *cno* and *disk* specify the device. The default value for *cno* is 0. Do not change this value unless instructed to do so by Sybase Technical Support.

-**s** *size*
is the size of the master device in 2K blocks. There are 512 2K blocks in 1MB. For example, a size of "15360" creates a 30MB master device. **buildmaster** verifies that the value you specify for this parameter does not exceed the space available to the master device, unless you use the **-m** parameter. See the installation or configuration documentation for your platform for more information about the size of master devices.

-**m**
rewrites only the *master* database, without initializing the master device. Use this parameter when the *master* database is corrupted but the other databases on the master device are undamaged.

-**r** *mastermirror*
starts the mirror of the master device.

-**q**
does not clear unallocated pages in *master* and *model* databases. This parameter has no effect when used with **-x**.

-x

rewrites only the *model* database, without changing or initializing the master device. Use this parameter when the *model* database is corrupted and you cannot load it successfully from a backup. If you modified *model*, you must restore it from a backup after re-initializing it with this parameter.

-v

prints the version number and copyright message for **buildmaster** and then exits.

Examples

```
buildmaster -d /dev/rsd1f -s8704
```

Initializes the raw device */dev/rsd1f* as a 17MB master device and creates the system databases *master*, *model*, and *tempdb* on the device.

Usage

- The **buildmaster** program initializes the specified database device as an Adaptive Server master device and builds the *master* and *model* databases on it.
- Use **buildmaster** only when Adaptive Server is shut down.
- The Adaptive Server installation program runs **buildmaster** and builds an initial *master* database on the database device you specify in answer to the program's prompts.
- If you run **buildmaster** with no parameters, it prompts for the following information:

  ```
  master disk name?
  master disk controller number?
  master disk size?
  configuration only? (y or n)
  databases only? (y or n)
  You must enter a response for each prompt.
  ```

  Answer "n" to "configuration only?", and use the following command to restore the configuration settings:

```
sp_configure "configuration file", 0, restore, file_name
```

  Answering "yes" to the "databases only?" prompt is the same as running buildmaster with the **-m** flag.
- For information on backing up and recovering system databases, see the *System Administration Guide*.
- The password to the default "sa" account reverts to null after you run **buildmaster -m**, and the account is unlocked. Loading a backup of the *master* database restores the "sa" password and account lock state to what they were when the dump was taken.

See also

Utility	**startserver**

charset

Description

Loads the character sets and sort order files.

Syntax

```
charset
 [-P password]
 [-S server]
 [-I interface]
 sort_order | charset
```

or

```
charset -v
```

Parameters

-P*password*

specifies your password. If you do not specify **-P**, **charset** prompts for your password.

-S*server*

specifies the name of the server on which to change the character set and sort order.

-I*interface*

specifies the network interface used by the server.

sort_order

specifies the name of the sort order file Adaptive Server will use.

charset

specifies the character set Adaptive Server will use.

-v

displays the version number and copyright message for **charset.**

Usage

- Before using **charset** you must set your SYBASE environment variable to point to the current release directory.

Permissions

You must be a System Administrator to use **charset.**

See also

Command	set
Utility	**langinstall**

dataserver

Description

The executable form of the Adaptive Server program.

Syntax

dataserver -d*devicename*
[-c*configurationfile*]
[-e*errorlogfile*]
[-m]
[-r*mastermirror*]
[-M*sharedmem_directory*]
[-i*interfaces_file_directory*]
[-s*servername*]
[-p*sso_login_name*]

or

dataserver -v

Parameters

-**d** *devicename*
is the full path name of the device for the *master* database. The master database device must be writable by the user who starts Adaptive Server. The default *master* database device name is *d_master*.

-**c***configurationfile*
specifies the full path name of an Adaptive Server configuration file. Use this parameter to start Adaptive Server with the configuration values in the specified configuration file. For more information, see the *System Administration Guide* .

-**e***errorlogfile*
is the full path name of the error log file for Adaptive Server system-level error messages.

-**m**
starts Adaptive Server in single-user mode.

-**r***mastermirror*
starts the mirror of the master device. Use this parameter to start Adaptive Server if the master device has been damaged.

-**M***sharedmem_directory*
places shared memory files in the specified directory instead of in the default location, *$SYBASE*. If *sharedmem_directory* starts with "/", the directory name is assumed to be absolute. Otherwise, the directory name is interpreted relative to *$SYBASE*.

-**i***interfaces_file_directory*
specifies the directory location of the interfaces file to search when connecting Adaptive Server. If **-i** is omitted, **dataserver** looks for a file named *interfaces* in the directory pointed to by your SYBASE environment variable.

-**s***servername*
specifies the name of the Adaptive Server to start. If **-s** is omitted, a server named SYBASE is started.

-**p***sso_login_name*
specifies the login name of a System Security Officer when starting Adaptive Server, for the purposes of getting a new password for that account. Adaptive Server generates a random password, displays it, encrypts it, and saves it in *master..syslogins* as that account's new password.

-**v**
prints the version number and copyright message for **dataserver** and then exits.

Usage

- Start Adaptive Server with the **startserver** command rather than by directly executing the **dataserver** program. If you need to change any of the default values, edit the *RUN_servername* file in your Sybase installation directory. See the **startserver** reference page for details.
- When you start an Adaptive Server with the **dataserver** program, Adaptive Server derives its running environment from:
  - the configuration file you specify in -**c** *configurationfile*;
  - the default configuration file, *servername.cfg*, if you did not specify the **-c** parameter;
  - default values if you did not specify either -**c** *configurationfile* or *servername.cfg*.
- For more information on these configuration parameters, see Chapter 17, "Setting Configuration Parameters," in the *System Administration Guide*.
- Because Adaptive Server passwords are encrypted, you cannot recover forgotten passwords. If all System Security Officers lose their passwords, the -**p** parameter generates a new password for a System Security Officer account. Start Adaptive Server with **-p**, immediately log into Adaptive Server with the new random password, and execute **sp_password** to reset your password to a more secure one.

- After you have finished running the Adaptive Server installation program, be sure to set the file permissions on the **dataserver** executable to limit who can execute it.
- If you do not specify an Adaptive Server name with the **-s** parameter, and you have not set the DSLISTEN environment variable, **dataserver** uses the default Adaptive Server name SYBASE. The value of the DSLISTEN environment variable overrides this default value, and the **-s** parameter overrides both the default and the DSLISTEN environment variable.

See also

Commands	disk mirror, disk remirror

defncopy

Description

Copies definitions for speccified views, rules, defaults, triggers, or procedures from a database to an operating system file or from an operating system file to a database.

Note The defncopy utility cannot copy table definitions or reports created with Report Workbench™.

Syntax

```
defncopy
 [-X]
 [-a display_charset]
 [-I interfaces_file]
 [-J [client_charset]]
 [-P password]
 [-R remote_server_principal]
 [-S [server]]
 [-U username
[-z language]
 {in filename dbname | out filename dbname
          [owner.]objectname [[owner.]objectname...] }
```

or

```
defncopy -v
```

Parameters

-X

initiates the login with client-side password encryption in this connection to the server. **defncopy** (the client) specifies to the server that password encryption is desired. The server sends back an encryption key, which **defncopy** uses to encrypt your password, and the server uses the key to authenticate your password when it arrives.

If **defncopy** crashes, the system creates a core file which contains your password. If you did not use the encryption option, the password appears in plain text in the file. If you used the encryption option, your password is not readable.

-**a** *display_charset*
runs **defncopy** from a terminal whose character set differs from that of the machine on which **defncopy** is running. Use **-a** in conjunction with **-J** to specify the character set translation file (*.xlt* file) required for the conversion. Use **-a** without **-J** only if the client character set is the same as the default character set.

Note The asscii_7 character set is compatible with all character sets. If either the Adaptive Server character set or the client character set is set to ascii_7, any 7-bit ASCII character can pass unaltered between client and server. Other characters produce conversion errors. For more information on character-set conversion, see the *System Administration Guide*.

-**I** *interfaces_file*
specifies the name and location of the interfaces file to search when connecting to Adaptive Server. If you do not specify **-I**, **defncopy** looks for a file named *interfaces* in the directory specified by the SYBASE environment variable.

-**J** *client_charset*
specifies the character set to use on the client. A filter converts input between *client_charset* and the Adaptive Server character set.

-**J** *client_charset* requests that Adaptive Server convert to and from *client_charset*, the client's character set.

-**J** with no argument sets character set conversion to NULL. No conversion takes place. Use this if the client and server are using the same character set.

Omitting **-J** sets the character set to a default for the platform. The default may not be the character set that the client is using. For more information about character sets and their associated flags, see the *System Administration Guide* .

-**P** *password*
specifies your password. If you do not specify **-P**, **defncopy** prompts for your password.

-**R** *remote_server_principal*
specifies the principal name for the server. By default, a server's principal name matches the server's network name (which is specified with the **-S** parameter or the DSQUERY environment variable). Use the **-R** parameter when the server's principal name and network name are not the same.

-**S** *server*
specifies the name of the Adaptive Server to which to connect. If you specify **-S** with no argument, **defncopy** looks for a server named SYBASE. If you do not specify **-S**, **defncopy** uses the server specified by your DSQUERY environment variable.

-**U** *username*
specifies a login name. Login names are case sensitive. If you do not specify *username*, **defncopy** uses the current user's operating system login name.

-**z** *language*
is the official name of an alternate language that the server uses to display **defncopy** prompts and messages. Without the **-z** flag, **defncopy** uses the server's default language. Add languages to an Adaptive Server at installation, or afterwards with the utility **langinstall** or the stored procedure **sp_addlanguage**.

in | out
specifies the direction of definition copy.

filename
specifies the name of the operating system file destination or source for the definition copy. The copy out overwrites any existing file.

dbname
specifies the name of the database to copy the definitions from or to.

owner
is optional if you or the Database Owner own the table being copied. If you do not specify an owner, **defncopy** first looks for a table of that name that you own, and then looks for one owned by the Database Owner. If another user owns the table, you must specify the owner name or the command fails.

objectname
specifies name(s) of database object(s) for **defncopy** to copy out. Do not use *objectname* when copying definitions in.

-**v**
displays the version and copyright message of **defncopy** and returns to the operating system.

Examples

```
defncopy -Usa -P -SMERCURY in new_proc stagedb
```

Copies definitions from the file *new_proc* into the database *stagedb* on server MERCURY. The connection with MERCURY is established with a user of name "sa" and a NULL password.

```
defncopy -S -z french out dc.out employees sp_calccomp
```

```
sp_vacation
```

Copies definitions for objects "*sp_calccomp*" and "*sp_vacation*" from the "*employees*" database on the SYBASE server to the file *dc.out*. Messages and prompts are displayed in "french". The user is prompted for a password.

Usage

- Invoke the **defncopy** program directly from the operating system. **defncopy** provides a non-interactive way of copying out definitions (**create** statements) for views, rules, defaults, triggers, or procedures from a database to an operating system file. Alternatively, it copies in all the definitions from a specified file.
- You must have **select** permission on the *sysobjects* and *syscomments* tables to copy out definitions; you do not need permission on the object itself.

  You may not have **select** permission on the *text* column of the *syscomments* table if the System Security Officer has reset the **allow select on syscomments.text column** parameter with the system procedure **sp_configure**. This reset restricts **select** permission to the object owner and the System Administrator. This restriction is required in order to run Adaptive Server in the *evaluated configuration*, as described in the installation and configuration documentation for your platform. In this case, the object owner or a System Administrator must execute **defncopy** to copy out definitions.

  Note If the text has been encrypted, it may be hidden from you even if you have all the required permissions. For more information, see "Verifying and Encrypting Source Text" in the *Transact-SQL User's Guide.*

- You must have the appropriate **create** permission for the type of object you are copying in. Objects copied in belong to the copier. A System Administrator copying in definitions on behalf of a user must log in as that user to give the user proper access to the reconstructed database objects.
- The **in** *filename* or **out** *filename* and the database name are required and must be stated unambiguously. For copying out, use file names that reflect both the object's name and its owner.
- **defncopy** ends each definition that it copies out with the comment:

  ```
  /* ### DEFNCOPY: END OF DEFINITION  */
  ```

  Definitions created as text must end with this comment so that **defncopy** can copy them in successfully.

- Enclose values specified to **defncopy** in quotation marks, if they contain characters that could be significant to the shell.

Warning! Long comments (more than 100 characters) placed before a **create** statement may cause **defncopy** to fail.

See also

System procedures	sp_checkreswords, sp_procqmode, sp_remap

defncopy_dce

Description

Copies definitions for specified views, rules, defaults, triggers, or procedures from a database to an operating system file or from an operating system file to a database.

Note The defncopy_dce utility cannot copy table definitions or reports created with Report Workbench™.

Syntax

```
defncopy_dce
 [-X]
 [-a display_charset]
 [-I interfaces_file]
 [-J [client_charset]]
 [-K keytab_file]
 [-P password]
 [-R remote_server_principal]
 [-S [server]]
 [-U username]
 [-V [security_options]]
 [-z language]
 [-Z security_mechanism]
 {in filename dbname | out filename dbname
          [owner.]objectname [[owner.]objectname...] }
```

or

```
defncopy_dce [-v]
```

Parameters

-a *display_charset*

runs **defncopy_dce** from a terminal whose character set differs from that of the machine on which **defncopy_dce** is running. **-a** in conjunction with **-J** specifies the character set translation file (*.xlt* file) required for the conversion. Use **-a** without **-J** only if the client character set is the same as the default character set.

Note The ascii_7 character set is compatible with all character sets. If either the Adaptive Server character set or the client character set is set to ascii_7, any 7-bit ASCII character can pass unaltered between client and server. Other characters produce conversion errors. For more information on character-set conversion, see the *System Administration Guide*.

-**I** *interfaces_file*
specifies the name and location of the interfaces file to search when connecting to Adaptive Server. If you do not specify **-I**, **defncopy_dce** looks for a file named *interfaces* in the directory specified by the SYBASE environment variable.

-**J** *client_charset*
specifies the character set to use on the client. A filter converts input between *client_charset* and the Adaptive Server character set.

-**J** *client_charset* requests that Adaptive Server convert to and from *client_charset*, the client's character set.

-**J** with no argument sets character set conversion to null. No conversion takes place. Use this if the client and server are using the same character set.

Omitting **-J** sets the character set to a default for the platform. The default may not be the character set that the client is using. For more information about character sets and their associated flags, see the *System Administration Guide*.

-**K** *keytab_file*
can be used only with DCE security. It specifies a DCE keytab file that contains the security key for the user name specified with **-U** parameter. Keytab files can be created with the DCE **dcecp** utility — see your DCE documentation for more information.

If the **-K** parameter is not supplied, the user of **defncopy_dce** must be logged in to DCE with the same username as specified with the **-U** parameter.

-**P** *password*
specifies your password. If you do not specify **-P**, **defncopy_dce** prompts for your password.

-**R** *remote_server_principal*
specifies the principal name for the server. By default, a server's principal name matches the server's network name (which is specified with the **-S** parameter or the DSQUERY environment variable). Use the **-R** parameter when the server's principal name and network name are not the same.

-**S** *server*
specifies the name of the Adaptive Server to which to connect. If you specify **-S** with no argument, **defncopy_dce** looks for a server named SYBASE. If you do not specify **-S**, **defncopy_dce** uses the server specified by your DSQUERY environment variable.

-**U** *username*
specifies a login name. Login names are case sensitive. If you do not specify *username*, **defncopy_dce** uses the current user's operating system login name.

-**V** *security_options*
specifies network-based user authentication. When you specify the **-V** parameter, users must log in to the network's security system before they can run **defncopy_dce**. If you specify the **-V** parameter, use the **-U** parameter to specify your network *username*. **defncopy_dce** ignores any password you supply with the **-P** parameter.

You can use the following security_options key letters to enable additional security services:

Table 5-10: Security Options for defncopy_dce

Key Letter	Security Service
c	Enable data confidentiality service
i	Enable data integrity service
m	Enable mutual authentication for establishing connections
o	Enable data origin stamping service
r	Enable data replay detection
q	Enable out-of-sequence detection

-**X**
initiates the login with client-side password encryption in this connection to the server. **defncopy_dce** (the client) specifies to the server that password encryption is desired. The server sends back an encryption key, which **defncopy_dce** uses to encrypt your password, and the server uses the key to authenticate your password when it arrives.

If **defncopy_dce** crashes, the system creates a core file which contains your password. If you did not use the encryption option, the password appears in plain text in the file. If you used the encryption option, your password is not readable.

-**Z** *security_mechanism*
specifies the name of a security mechanism to use on the connection.

Security mechanism names are defined in the *$SYBASE/install/libctl.cfg* configuration file. If no security mechanism name is supplied, the default mechanism is used. For more information on security mechanism names, see the description of the *libctl.cfg* file in the *Open Client/Server Configuration Guide for UNIX*.

-z *language*
is the official name of an alternate language that the server uses to display **defncopy_dce** prompts and messages. Without the **-z** flag, **defncopy_dce** uses the server's default language. Add languages to a Adaptive Server at installation, or afterwards with the utility **langinstall** or the stored procedure **sp_addlanguage**.

in | out
specifies the direction of definition copy.

filename
specifies the name of the operating system file destination or source for the definition copy. The copy out overwrites any existing file.

dbname
specifies the name of the database to copy the definitions from or to.

objectname
specifies name(s) of database object(s) for **defncopy_dce** to copy out. Do not use *objectname* when copying definitions in.

-v
displays the version and copyright message of **defncopy_dce** and returns to the operating system.

Examples

```
defncopy_dce -Usa -P -SMERCURY in new_proc stagedb
```

Copies definitions from the file *new_proc* into the database *stagedb* on server MERCURY. The connection with MERCURY is established with a user of name "sa" and a NULL password.

```
defncopy_dce -S -z french out dc.out employees
sp_calccomp  sp_vacation
```

Copies definitions for objects "*sp_calccomp*" and "*sp_vacation*" from the "*employees*" database on the SYBASE server to the file *dc.out*. Messages and prompts are displayed in "french". The user is prompted for a password.

Usage

- **defncopy_dce** is available only for the IBM RS/6000 platform of UNIX.
- Invoke the **defncopy_dce** program directly from the operating system. **defncopy_dce** provides a non-interactive way of copying out definitions (**create** statements) for views, rules, defaults, triggers, or procedures from a database to an operating system file. Alternatively, it copies in all the definitions from a specified file.
- You must have **select** permission on the *sysobjects* and *syscomments* tables to copy out definitions; you do not need permission on the object itself.

- You may not have **select** permission on the *text* column of the *syscomments* table if the System Security Officer has reset the **allow select on syscomments.text column** parameter with the system procedure **sp_configure**. This reset restricts **select** permission to the object owner and the System Administrator. This restriction is required in order to run Adaptive Server in the *evaluated configuration*, as described in the installation and configuration documentation for your platform. In this case, the object owner or a System Administrator must execute **defncopy_dce** to copy out definitions. You must have the appropriate **create** permission for the type of object you are copying in. Objects copied in belong to the copier. A System Administrator copying in definitions on behalf of a user must log in as that user to give the user proper access to the reconstructed database objects.
- The **in** *filename* or **out** *filename* and the database name are required and must be stated unambiguously. For copying out, use file names that reflect both the object's name and its owner.
- **defncopy_dce** ends each definition that it copies out with the comment:

  ```
  /* ### defncopy_dce: END OF DEFINITION  */
  ```

  Definitions created as text must end with this comment so that **defncopy_dce** can copy them in successfully.
- Enclose values specified to **defncopy_dce** in quotation marks, if they contain characters that could be significant to the shell.

Warning! Long comments (more than 100 characters) placed before a **create** statement may cause **defncopy_dce** dce to fail.

See also

System procedures	sp_checkreswords, sp_procqmode, sp_remap

dscp

Description

Allows you to view and edit server entries in the interfaces file from the command line.

Syntax

dscp [-p]

or

dscp -v

To exit from **dscp**: *quit* or *exit*

Parameters

-**p**
suppresses command-line prompts.

-**v**
displays the version and copyright message of **dscp** and returns to the operating system.

Examples

```
dscp -p
```

Opens the default interfaces file for editing and suppresses the command-line prompt.

Usage

- The **dscp** utility program is a text-based utility.
- For more information about the **dscp** utility program, see Chapter 4, “Using dscp.”

See also

Utility	**dsedit**

dscp_dce

Description

Allows you to view and edit server entries in the DCE directory service file in command line mode.

Syntax

dscp_dce [-p] [quit | exit]

or

dscp_dce [-v]

Parameters

-p
suppresses command-line prompts.

quit, **exit**
closes the specified interfaces file and exits the session.

-v
displays the version and copyright message of **dscp_dce** and returns to the operating system.

Usage

- **dscp_dce** is available only for the IBM RS/6000 platform of UNIX.
- You must have a DCE account to run **dscp_dce**.
- For more information about the **dscp_dce** utility programs, see Chapter 4, “Using dscp.”

See also

Utility	**dsedit_dce**

dsedit

Description

Allows you to view and edit server entries in the interfaces file using a graphical user interface based on X11/Motif.

Syntax

```
dsedit
```

or

```
dsedit -v
```

Parameters

-**v**
displays the version and copyright message of **dsedit**.

Usage

- You must set the SYBASE environment variable to the location of the current version of Adaptive Server before you can use **dsedit**.
- You must set the DISPLAY environment variable before invoking **dsedit**, unless you are only using the **-v** parameter to display the version number.
- For more information about the **dsedit** utility program, see Chapter 3, “Using dsedit”.

See also

Utility	**dscp**

dsedit_dce

Description

Creates and modifies network connection information in the DCE directory service file.

Syntax

dsedit_dce

or

dsedit_dce [-v]

Parameters

-v

displays the version and copyright message of **dsedit_dce**.

Usage

- **dsedit_dce** is available only for the IBM RS/6000 platform of UNIX.
- **dsedit_dce** creates and modifies network connection information in the DCE directory service file.
- You must set the SYBASE environment variable to the location of the current version of Adaptive Server before you can use **dsedit_dce**.
- If you do not use the optional **-v** parameter, be sure to set the DISPLAY environment variable.
- To use **dsedit_dce**, you must be logged into DCE.
- For more information about the **dsedit_dce** utility programs, see Chapter 3, “Using dsedit.”

See also

Utility	**dscp_dce**

extractjava

Description

Copies a retained JAR and the classes it contains from an Adaptive Server into a client file.

Refer to *Java in Adaptive Server Enterprise* for more information about how this utility is used when Java is enabled in the database.

Syntax

extractjava
-j *jar_name*
-f *file_name*
[-S *server*]
[-U *username*]
[-P *password*]
[-D *database*]
[-I *interfaces_file*]
[-a *display_charset*]
[-J *client_charset*]
[-z *language*]
[-t *timeout*] [-v]

Parameters

-j *jar_name*
the name assigned to the retained JAR in the database that is the source of the transfer.

-f *file_name*
the name of the client file that is the target of the transfer.

-S *server*
the name of the server.

-U *user_name*
an Adaptive Server login name. If you omit the **–U** flag and parameter, or if you specify the **–U** flag with no parameter, Adaptive Server uses the current user's operating system login name.

-P *password*
an Adaptive Server password. If you omit the **–P** flag and parameter, **extractjava** prompts for a password. If you specify the **–P** flag with no password, the null password is used.

-D *database*
the name of the database in which to install the JAR. If you omit the **–D** flag, or if you specify the **–D** flag with no parameter, the user's default database is used.

-I *interfaces_file*
the name and location of the interfaces file to search when connecting to Adaptive Server. If you omit the **–I** flag and parameter, or if you specify the **–I** flag with no parameter, the interfaces file in the directory designated by your SYBASE environment variable is used.

-a *display_charset*
allows you to use **extractjava** from a machine where the character set differs that of the server. Use **-a** in conjunction with **-J** to specify the character set translation file (.xlt file) required for the conversion. Use **-a** without **-J** only if the client character set is the same as the default character set

-J *client_charset*
specifies the character set to use on the client. **extractjava** uses a filter to convert input between client_charset and the Adaptive Server character set.

-J client_charset requests that Adaptive Server convert to and from client_charset, the character set used on the client.

-J with no argument disables character set conversion. Use this if the client and server use the same character set.

Omitting **-J** sets the character set to a default for the platform, which may not necessarily be the character set that the client is using. See the *System Administration Guide* for more information about character sets and associated flags.

-z *language*
the name of an alternate language for displaying **extractjava** prompts and messages. Without the **-z** flag, extractjava uses the server's default language. You can add languages to an Adaptive Server during installation or afterward, using the **langinstall** utility or the **sp_addlanguage** stored procedure.

-t *timeout*
specifies the number of seconds before a SQL command times out. If you do not specify a timeout, the command runs indefinitely. This affects commands issued from within **extractjava**, not the connection time. The default timeout for logging into **extractjava** is 60 seconds.

-v
prints the version number and copyright message for **extractjava** and then exits.

Examples

```
extractjava -j employees -f '/home/usera/jars/addr.jar'
-new
```

Downloads the classes associated with the employees JAR to the client file *newaddr.jar*.

Usage

- If the target client file already exists, **extractjava** overwrites its contents.
- The parameter flags **-f**, **-j**, **-S**, **-U**, **-P**, **-D**, and **–I** can be written with or without a space between the flag letter and the following parameter.
- When you execute **extractjava**, an exclusive lock is placed on *sysxtypes*.
- If **-jar** is specified, an exclusive table lock is placed on *sysjars*.

Permissions

You need to be a System Administrator or Database Owner to use **extractjava**.

See also

Commands	remove java
System procedures	sp_helpjava
System tables	*sysjars*, *sysxtypes*
Utilities	**installjava**

installjava

Description

Installs a JAR from a client file into an Adaptive Server.

Refer to *Java in Adaptive Server Enterprise* for more information about how this utility is used when Java is enabled in the database.

Syntax

installjava
-f *file_name*
[-new | -update]
[-j *jar_name*]
[-S *server*]
[-U *username*]
[-P *password*]
[-D *database*]
[-I *interfaces_file*]
[-a *display_charset*]
[-J *client_charset*]
[-z *language*]
[-t *timeout*] [-v]

Parameters

-f *file_name*
the name of the source file containing the classes to be installed in the database.

new | update
specifies whether the classes in the file already exist in the database.

If you specify the new parameter, you cannot install a class with the same name as an existing class.

If you specify the update parameter, you can install a class with the same name as an existing class, and the newly installed class replaces the existing class.

-j *jar_name*
the name of the JAR containing the classes to be installed in the database. Indicates that the JAR file should be saved in the database and associated with the classes it contains.

-S *server*
the name of the server.

-U *user_name*
an Adaptive Server login name. If you omit the **–U** flag and parameter, or if you specify the **–U** flag with no parameter, Adaptive Server uses the current user's operating system login name.

-P *password*
an Adaptive Server password. If you omit the **–P** flag and parameter, **installjava** prompts for a password. If you specify the **–P** flag with no password, the null password is used.

-D *database*
the name of the database in which to install the JAR. If you omit the **–D** flag, or if you specify the **–D** flag with no parameter, the user's default database is used.

-I *interfaces_file*
the name and location of the interfaces file to search when connecting to Adaptive Server. If you omit the **–I** flag and parameter, or if you specify the **–I** flag with no parameter, the interfaces file in the directory designated by your SYBASE environment variable is used.

-a *display_charset*
allows you to use **installjava** from a machine where the character set differs that of the server. Use **-a** in conjunction with **-J** to specify the character set translation file (.xlt file) required for the conversion. Use **-a** without **-J** only if the client character set is the same as the default character set.

-J *client_charset*
apecifies the character set to use on the client. **installjava** uses a filter to convert input between client_charset and the Adaptive Server character set.

-J client_charset requests that Adaptive Server convert to and from client_charset, the character set used on the client.

-J with no argument disables character set conversion. Use this if the client and server use the same character set.

Omitting **-J** sets the character set to a default for the platform, which may not necessarily be the character set that the client is using. See the *System Administration Guide* for more information about character sets and associated flags.

-z *language*
the name of an alternate language for displaying **installjava** prompts and messages. Without the -z flag, installjava uses the server's default language. You can add languages to an Adaptive Server during installation or afterward, using the **langinstall** utility or the **sp_addlanguage** stored procedure.

-t *timeout*
specifies the number of seconds before a SQL command times out. If you do not specify a timeout, the command runs indefinitely. This affects commands issued from within **instjallava**, not the connection time. The default timeout for logging into **installjava** is 60 seconds.

-v
prints the version number and copyright message for **installjava** and then exits.

Examples

```
installjava -f '/home/usera/jars/addr.jar' -new
```

Installs *addr.jar* and its classes, but does not retain the association between the JAR and classes.

```
installjava -f '/home/usera/jars/addr.jar' -update
   -j employees
```

Reinstalls *addr.jar* and associates its classes with the employees JAR name.

Usage

- Any user can reference installed classes.
- The parameter flags **-f**, **-j**, **-S**, **-U**, **-P**, **-D**, and **–I** can be written with or without a space between the flag letter and the following parameter.

Adding New JARs

- If you use new with the -jar option and a JAR of that name already exists in the database, an exception is raised.
- If any classes of the same name as those in the source JAR already exist in the database, an exception is raised.

Updating JARs and Classes

Warning! If you alter a class used as a column datatype by reinstalling a modified version of the class, you must make sure that the modified class can read and use existing objects (rows) in tables using that class as a datatype. Otherwise, you may be unable to access those objects without reinstalling the class.

- If you use **-update** with the **-jar** option:
  - All classes in the database associated with the target JAR are deleted from the database and the classes in the source JAR file installed in their place.

    - If a class in the source JAR file is already installed in the database but is not attached to a JAR, the class in the source JAR is installed in the database and the unattached class is deleted.
- If you use **-update** without the **-jar** option:
    - Classes in the source JAR file replace unattached classes of the same name.
    - Classes in the source JAR that do not correspond to an installed class are installed as unattached classes in the database.

Locks

- When you execute **installjava**, an exclusive lock is placed on *sysxtypes*.
- If **-jar** is specified, an exclusive table lock is placed on *sysjars*.

Permissions

You need to be a System Administrator or Database Owner to use **installjava**.

See also

Commands	remove java
System procedures	sp_helpjava
System tables	*sysjars*, *sysxtypes*
Utilities	**extractjava**

isql

Description

Interactive SQL parser to Adaptive Server.

Syntax

isql [-b] [-e] [-F] [-n] [-p] [-X] [-Y]
[-a *display_charset*]
[-A *size*]
[-c *cmdend*]
[-D *database*]
[-E *editor*]
[-h *headers*]
[-H *hostname*]
[-i *inputfile*]
[-I *interfaces_file*]
[-J *client_charset*
[-l *login_timeout*]
[-m *errorlevel*]
[-o *outputfile*]
[-P *password*]
[-R *remote_server_principal*]
[-s *colseparator*]
[-S *server*]
[-t *timeout*]
[-U *username*]
[-w *columnwidth*]
[-z *language*]

or

isql -v

To terminate a command: **go**

To clear the query buffer: **reset**

To call the default editor: **vi**

To execute an operating system command: **!! *command***

To exit from **isql**: **quit or exit**

Parameters

-**b**
disables the display of the table headers output.

-**e**
echoes input.

-**F**
enables the FIPS flagger. When you specify the **-F** parameter, the server returns a message when it encounters a non-standard SQL command. This option does not disable SQL extensions. Processing completes when you issue the non-ANSI SQL command.

-n
removes numbering and the prompt symbol (>) from the echoed input lines in the output file when used in conjunction with **-e**.

-p
prints performance statistics.

-X
initiates the login connection to the server with client-side password encryption. **isql** (the client) specifies to the server that password encryption is desired. The server sends back an encryption key, which **isql** uses to encrypt your password, and the server uses the key to authenticate your password when it arrives.

If **isql** crashes, the system creates a core file that contains your password. If you did not use the encryption option, the password appears in plain text in the file. If you used the encryption option, your password is not readable.

-Y
tells the Adaptive Server to use chained transactions.

-a *display_charset*
runs **isql** from a terminal whose character set differs from that of the machine on which **isql** is running. Use **-a** in conjunction with **-J** to specify the character set translation file (*.xlt* file) required for the conversion. Use **-a** without **-J** only if the client character set is the same as the default character set.

Note The ascii_7 character set is compatible with all character sets. If either the Adaptive Server character set or the client character set is set to ascii_7, any 7-bit ASCII character can pass unaltered between client and server. Other characters produce conversion errors. F or more information on character set conversion, see the *System Administration Guide*.

-A *size*
specifies the network packet size to use for this **isql** session For example:

```
isql -A 2048
```

sets the packet size to 2048 bytes for this **isql** session. To check your network packet size, enter:

```
select * from sysprocesses
```

The value is displayed under the *network_pktsz* heading.

size must be between the values of the **default network packet size** and **maximum network packet size** configuration parameters, and must be a multiple of 512.

Use larger-than-default packet sizes to perform I/O-intensive operations, such as **readtext** or **writetext** operations.

Setting or changing Adaptive Server's packet size does not affect the packet size of remote procedure calls.

-c *cmdend*
changes the command terminator. By default, you terminate commands and send them to by typing "**go**" on a line by itself. When you change the command terminator, do not use SQL reserved words or control characters.

-D *database*
selects the database in which the **isql** session begins.

-E *editor*
specifies an editor other than the default editor **vi**.

-h *headers*
specifies the number of rows to print between column headings. The default prints headings only once for each set of query results.

-H *hostname*
sets the client hostname.

-i *inputfile*
specifies the name of the operating system file to use for input to **isql**. The file must contain command terminators ("go" is the default).

Specifying the parameter as follows: **-i *inputfile***

is equivalent to: **< *inputfile***

If you use **-i** and do not specify your password on the command line, **isql** prompts you for it.

If you use **<** *inputfile* and do not specify your password on the command line, you must specify your password as the first line of the input file.

-I *interfaces_file*
specifies the name and location of the interfaces file to search when connecting to Adaptive Server. If you do not specify **-I**, **isql** looks for a file named *interfaces* in the directory specified by your SYBASE environment variable.

-J *client_charset*
specifies the character set to use on the client. **-J** *client_charset* requests that Adaptive Server convert to and from *client_charset*, the character set used on the client. A filter converts input between *client_charset* and the Adaptive Server character set.

-J with no argument sets character set conversion to NULL. No conversion takes place. Use this if the client and server use the same character set.

Omitting **-J** sets the character set to a default for the platform. The default may not necessarily be the character set that the client is using. F or more information about character sets and the associated flags, see Chapter 20, "Configuring Client/Server Character Set Conversions," in the *System Administration Guide*. Table 5-11 lists platform defaults.

Table 5-11: Default character sets for different platforms

Platform	Default Character Set
Sun Solaris, Digital UNIX, Pyramid, NCR, RS/6000	iso_1
HP-UX	roman8
OS/2, Novell NetWare 386	cp850
Macintosh	mac

-l *login_timeout*
specifies the maximum timeout value allowed when connecting to Adaptive Server. The default is 60 seconds. This value affects only the time that **isql** waits for the server to respond to a login attempt. To specify a timeout period for command processing, use the **-t** *timeout* parameter.

-m *errorlevel*
customizes the error message display. For errors of the severity level specified or higher, only the message number, state, and error level are displayed; no error text appears. For error levels lower than the specified level, nothing appears.

-o *outputfile*
specifies the name of an operating system file to store the output from **isql**. Specifying the parameter as follows: **-o *outputfile***

is similar to: **> *outputfile***

-**P** *password*
specifies your Adaptive Server password. If you do not specify the **-P** flag, **isql** prompts for a password. If your password is NULL, use the **-P** flag without any password.

-**R** *remote_server_principal*
specifies the principal name for the server as defined to the security mechanism. By default, a server's principal name matches the server's network name (which is specified with the **-S** parameter or the DSQUERY environment variable). Use the **-R** parameter when the server's principal name and network name are not the same.

-**s** *colseparator*
resets the column separator character, which is blank by default. To use characters that have special meaning to the operating system (for example, "|", ";", "&", "<", ">"), enclose them in quotes or precede them with a backslash.

-**S** *server*
specifies the name of the Adaptive Server to which to connect. **isql** looks this name up in the interfaces file. If you specify **-S** with no argument, **isql** looks for a server named SYBASE. If you do not specify **-S**, **isql** looks for the server specified by your DSQUERY environment variable.

-**t** *timeout*
specifies the number of seconds before a SQL command times out. If you do not specify a timeout, the command runs indefinitely. This affects commands issued from within **isql**, not the connection time. The default timeout for logging into **isql** is 60 seconds.

-**U** *username*
specifies a login name. Login names are case sensitive.

-**w** *columnwidth*
sets the screen width for output. The default is 80 characters. When an output line reaches its maximum screen width, it breaks into multiple lines.

-**z** *language*
is the official name of an alternate language to display **isql** prompts and messages. Without **-z**, **isql** uses the server's default language. You can add languages to an Adaptive Server during installation or afterward, using the **langinstall** utility or the **sp_addlanguage** stored procedure.

-**v**
prints the version number and copyright message for **isql** and then exits.

Examples

```
isql -Ujoe -Pabracadabra
```

```
1> select *
2> from authors
3> where city = "Oakland"
4> vi
```

Puts you in a text file where you can edit the query. When you write and save the file, you are returned to **isql**. The query appears; type "go" on a line by itself to execute it.

```
isql -Ualma
Password:
1> select *
2> from authors
3> where city = "Oakland"
4> reset
1> quit
```

reset clears the query buffer. **quit** returns you to the operating system.

```
isql -a mac -J roman8
```

Specifies that you are running **isql** from a Macintosh against a server that is using the roman8 character set.

Usage

- See Chapter 1, “Using the isql Utility” for details on **isql**.
- See the *Adaptive Server Reference Manual* for more information regarding **default network packet size** and **maximum network packet size** configuration parameters.
- The 5701 (“changed database”) server message is no longer displayed after login or issuing a **use database** command.
- There are two new optional flags:

  -**b** – disables column headers from printing
  -D *database* – selects the startup database that **isql** uses
- Error message format differs from previous versions of **isql**. If you have scripts that perform routines based on the values of these messages you may need to re-write them.
- To use **isql** interactively, give the command **isql** (and any of the optional parameters) at your operating system prompt. The **isql** program accepts SQL commands and sends them to Adaptive Server. The results are formatted and printed on standard output. Exit **isql** with **quit** or **exit.**

- Send a command to Adaptive Server by typing the command terminator (**go**, by default) at the beginning of a line. You can follow the command terminator with an integer to specify how many times to run the command. For example, to execute a command 100 times, type:

  ```
  select x = 1
  go 100
  ```

  The results display once at the end of execution.

- The **-y** *sybase_directory* parameter has been removed.

- If you enter an option more than once on the command line, **isql** uses the last value. For example, if you enter the following command:

  ```
  isql -c"." -csend
  ```

  "send", the second value for **-c**, overrides ".", the first value. This enables you to override any aliases you set up.

- To call an editor on the current query buffer, enter its name as the first word on a line. Define your preferred callable editor by specifying it with the EDITOR environment variable. If EDITOR is not defined, the default is **vi**.

  Execute operating system commands by starting a line with "!!" followed by the command. Call alternate editors this way, without defining EDITOR.

- To clear the existing query buffer, type **reset** on a line by itself. **isql** discards any pending input. You can also press Ctrl-c anywhere on a line to cancel the current query and return to the **isql** prompt.

- Read in an operating system file containing a query for execution by **isql** as follows:

  ```
  isql -U alma -Ppasswd < input_file
  ```

  The file must include a command terminator. The results appear on your terminal. Read in an operating system file containing a query and direct the results to another file as follows:

  ```
  isql -U alma -Ppasswd < input_file > output_file
  ```

- Case is significant for the **isql** flags.

- **isql** displays only 6 digits of float or real data after the decimal point, rounding off the remainder.

- When you are using **isql** interactively, read an operating system file into the command buffer with the command:

```
:r filename
```

Do not include the command terminator in the file; once you have finished editing, enter the terminator interactively on a line by itself.

- You can include comments in a Transact-SQL statement submitted to Adaptive Server by **isql**. Open a comment with "/*". Close it with "*/", as shown in the following example:

```
select au_lname, au_fname
/*retrieve authors' last and first names*/
from authors, titles, titleauthor
where authors.au_id = titleauthor.au_id
and titles.title_id = titleauthor.title_id
/*this is a three-way join that links authors
**to the books they have written.*/
```

If you want to comment out a **go** command, it should not be at the beginning of a line. For example:

```
/*
**go
*/
```

should be used to comment out the **go** command instead of:

```
/*
go
*/
```

See also

Commands	create schema, set
Datatype	Exact Numeric Datatypes
System ESP	xp_sendmail
System procedures	sp_addlanguage, sp_addlogin, sp_addremotelogin, sp_add_resource_limit, sp_bindexeclass, sp_configure, sp_defaultlanguage, sp_droplanguage, sp_helplanguage, sp_processmail, sp_remoteoption, sp_serveroption, sp_showcontrolinfo, sp_unbindexeclass, sp_volchanged

isql_dce

Description

Interactive SQL parser to Adaptive Server.

Syntax

isql_dce [-b] [-e] [-F] [-n] [-p] [-v] [-X] [-Y]
[-a *display_charset*]
[-A *size*]
[-c *cmdend*]
[-D *database*]
[-E *editor*]
[-h *headers*]
[-H *hostname*]
[-i *inputfile*]
[-I *interfaces_file*]
[-J *client_charset*]
[-K *keytab_file*]
[-l *login_timeout*]
[-m *errorlevel*]
[-o *outputfile*]
[-P *password*]
[-R *remote_server_principal*]
[-s *colseparator*]
[-S *server*]
[-t *timeout*]
[-U *username*]
[-w *columnwidth*]
[-z *language*]

or

isql_dce -v

To terminate a command: **go**

To clear the query buffer: **reset**

To call the default editor: **vi**

To execute an operating system command: **!!** ***command***

To exit from **isql_dce**: **quit** or **exit**

Parameters

-e
echoes input.

-**F**
enables the FIPS flagger. With this parameter, the server flags any non-SQL standard commands.

-**p**
prints performance statistics.

-**n**
removes numbering and the prompt symbol (>) from the echoed input lines in the OUTPUT file when used in conjunction with **-e**.

-**v**
prints the version and copyright of the **isql_dce** software you are using and then exits.

-**X**
initiates the login connection to the server with client-side password encryption. **isql_dce** (the client) specifies to the server that password encryption is desired. The server sends back an encryption key, which **isql_dce** uses to encrypt your password, and the server uses the key to authenticate your password when it arrives.

If **isql_dce** crashes, the system creates a core file that contains your password. If you did not use the encryption option, the password appears in plain text in the file. If you used the encryption option, your password is not readable.

-**Y**
tells the Adaptive Server to use chained transactions.

-**a** *display_charset*
runs **isql_dce** from a terminal whose character set differs from that of the machine on which **isql_dce** is running. **-a** in conjunction with **-J** specifies the character set translation file (*.xlt* file) required for the conversion. Use **-a** without **-J** only if the client character set is the same as the default character set.

Note The ascii_7 character set is compatible with all character sets. If either the Adaptive Server character set or the client character set is set to ascii_7, any 7-bit ASCII character can pass unaltered between client and server. Other characters produce conversion errors. For more information on character set conversion, see the *System Administration Guide*.

-**A** *size*
specifies the network packet size to use for this **isql_dce** session For example:

```
isql_dce -A 2048
```

sets the packet size to 2,048 bytes for this **isql_dce** session. To check, type:

```
select * from sysprocesses
```

The value is displayed under the *network_pktsz* heading.

size must be between the values of the **default network packet size** and **maximum network packet size** configuration variables, and must be a multiple of 512.

Use larger-than-default packet sizes to perform I/O-intensive operations, such as **readtext** or **writetext** operations.

Setting or changing the server packet size does not affect remote procedure call packet size.

-c *cmdend*
changes the command terminator. By default, you terminate commands and send them to Adaptive Server by typing "go" on a line by itself. When you change the command terminator, do not use SQL reserved words or control characters.

-D *database*
selects the database that the **isql_dce** session begins in.

-E *editor*
specifies an editor other than the default editor **vi**.

-h *headers*
specifies how many rows to print between column headings. The default prints headings only once for each set of query results.

-H *hostname*
sets the client hostname.

-i *inputfile*
specifies the name of the operating system file to use for input to **isql_dce**. The file must contain command terminators ("go" is the default).

Specifying the parameter as follows: **-i *inputfile***

is equivalent to: **< *inputfile***

If you use **-i** and do not specify your password on the command line, **isql_dce** prompts you for it.

If you use **<** *inputfile* and do not specify your password on the command line, you must specify your password as the first line of the input file.

-I *interfaces_file*
specifies the name and location of the interfaces file to search when connecting to Adaptive Server. Without **-I**, **isql_dce** looks for a file named *interfaces* in the directory specified by your SYBASE environment variable.

-**J** *client_charset*
specifies the character set to use on the client.
-J *client_charset* requests that Adaptive Server convert to and from *client_charset*, the character set used on the client. A filter converts input between *client_charset* and the Adaptive Server character set.

-**J** with no argument sets character set conversion to null. No conversion takes place. Use this if the client and server use the same character set.

Omitting **-J** sets the character set to a default for the platform. The default may not necessarily be the character set that the client is using. For more information about character sets and the associated flags, see the *System Administration Guide*. Table 5-12 lists platform defaults.

Table 5-12: Default character sets for different platforms

Platform	Default Character Set
Sun Solaris, Digital UNIX, Pyramid, NCR, RS/6000	iso_1
HP-UX	roman8
OS/2, Novell NetWare 386	cp850
Macintosh	mac

-**K***keytab_file*
can be used only with DCE security. It specifies a DCE keytab file that contains the security key for the user name specified with **-U** parameter. Keytab files can be created with the DCE **dcecp** utility—see your DCE documentation for more information.

If the -**K**
parameter is not supplied, the user of **isql_dce** must be logged in to DCE with the same username as specified with the **-U** parameter.

-**l** *login_timeout*
specifies the maximum timeout value allowed when connecting to Adaptive Server. The default is 60 seconds. This value affects only the time that **isql_dce** waits for the server to respond to a login attempt. To specify a timeout period for command processing, use the **-t** *timeout* parameter.

-**m** *errorlevel*
customizes the error message display. For errors of the severity level specified or higher, only the message number, state, and error level are displayed; no error text appears. For error levels lower than the specified level, nothing appears.

-**o** *outputfile*
specifies the name of an operating system file to store the output from **isql_dce**. Specifying the parameter as follows: **-o *outputfile*** is similar to: ***outputfile***

-**P** *password*
specifies your Adaptive Server password. If you do not specify the **-P** flag, **isql_dce** prompts for a password. If your password is NULL, use the **-P** flag without any password at the end of the command line.

-**s** *colseparator*
resets the column separator character, which is blank by default. To use characters that have special meaning to the operating system (for example, "|", ";", "&", "<", ">"), enclose them in quotes or precede them with a backslash.

-**S** *server*
specifies the name of the Adaptive Server to which to connect. **isql_dce** looks this name up in the interfaces file. If you specify **-S** with no argument, **isql_dce** looks for a server named SYBASE. Without **-S**, **isql_dce** looks for the server specified by your DSQUERY environment variable.

-**t** *timeout*
specifies the number of seconds before a SQL command times out. If you do not specify a timeout, the command runs indefinitely. This affects commands issued from within **isql_dce**, not the connection time. The default timeout for logging into **isql_dce** is 60 seconds.

-**U** *username*
specifies a login name. Logins are case sensitive.

-**w** *columnwidth*
sets the screen width for output. The default is 80 characters. When an output line reaches its maximum screen width, it breaks into multiple lines.

-**z** *language*
is the official name of an alternate language to display **isql_dce** prompts and messages. Without **-z**, **isql_dce** uses the server's default language. You can add languages to an Adaptive Server during installation or afterward, using the **langinstall** utility or the **sp_addlanguage** stored procedure.

-A *size*
specifies the network packet size to use for this **isql_dce** session For example:

```
isql_dce -A 2048
```

sets the packet size to 2048 bytes for this **isql_dce** session. *size* must be between the values of the **default network packet size** and **max network packet size** configuration parameters, one-third the size of the **additional network memory** configuration parameter, and it must be a multiple of 512.

-v
prints the version number and copyright message for **isql_dce** and then exits.

Examples

```
isql_dce -Ujoe -Pabracadabra
1> select *
2> from authors
3> where city = "Oakland"
4> vi
```

Puts you in a text file where you can edit the query. When you write and save the file, you are returned to **isql_dce**. The query appears; type "go" on a line by itself to execute it.

```
isql_dce -U alma
Password:
1> select *
2> from authors
3> where city = "Oakland"
4> reset
1> quit
```

reset clears the query buffer. **quit** returns you to the operating system.

```
isql_dce -a mac -J roman8
```

Specifies that you are running **isql_dce** from a Macintosh against a server that is using the roman8 character set.

Usage

- **isql_dce** is available only for the IBM RS/6000 platform of UNIX.
- See Chapter 1, “Using the isql Utility” for details on **isql_dce**.
- See the *Adaptive Server Reference Manual* for more information regarding **default network packet size** and **maximum network packet size** configuration parameters.

- To use **isql_dce** interactively, give the command **isql_dce** (and any of the optional flags) at your operating system prompt. The **isql_dce** program accepts SQL commands and sends them to Adaptive Server. The results are formatted and printed on standard output. Exit **isql_dce** with **quit** or **exit.**
- Send a command to Adaptive Server by typing the command terminator (**go**, by default) at the beginning of a line. You can follow the command terminator with an integer to specify how many times to run the command. For example, to execute a command 100 times, type:

  ```
  select x = 1
  go 100
  ```

  The results appear once at the end of execution.
- If you enter an option more than once on the command line, **isql_dce** uses the last value. For example, if you enter the following command:

  ```
  isql_dce -c"." -csend
  ```

  "send", the second value for **-c**, overrides ".", the first value. This enables you to override any aliases you set up.
- To call an editor on the current query buffer, enter its name as the first word on a line. Define your preferred callable editor by specifying it with the EDITOR environment variable. If EDITOR is not defined, the default is **vi**.

  Execute operating system commands by starting a line with "!!" followed by the command. Call alternate editors this way, without defining EDITOR.
- To clear the existing query buffer, type **reset** on a line by itself. **isql_dce** discards any pending input. You can also press Ctrl-c anywhere on a line to cancel the current query and return to the **isql_dce** prompt.
- Read in an operating system file containing a query for execution by **isql_dce** as follows:

  ```
  isql_dce -U alma -P****** < input_file
  ```

  The file must include a command terminator. The results appear on your terminal. Read in an operating system file containing a query and direct the results to another file as follows:

  ```
  isql_dce -U alma -P****** < input_file > output_file
  ```
- Case is significant for the **isql_dce** flags.

- **isql_dce** displays only 6 digits of float or real data after the decimal point, rounding off the remainder.
- When you are using **isql_dce** interactively, read an operating system file into the command buffer with the command:

  ```
  :r filename
  ```

  Do not include the command terminator in the file; once you have finished editing, enter the terminator interactively on a line by itself.
- You can include comments in a Transact-SQL statement that is submitted to Adaptive Server by **isql_dce**. For more information, see the *Adaptive Server Reference Manual*.

See also

Commands	create schema, set
Datatype	Exact Numeric Datatypes
System ESP	xp_sendmail
System procedures	sp_addlanguage, sp_addlogin, sp_addremotelogin, sp_add_resource_limit, sp_bindexeclass, sp_configure, sp_defaultlanguage, sp_droplanguage, sp_helplanguage, sp_processmail, sp_remoteoption, sp_serveroption, sp_showcontrolinfo, sp_unbindexeclass, sp_volchanged

langinstall

Description

Installs a new language in an Adaptive Server.

Syntax

```
langinstall
 [-S server]
 [-I interfaces_file]
 [-P password]
 [-R release_number]
 language character_set
```

or

```
langinstall -v
```

Parameters

-**S** *server*
specifies the name of the Adaptive Server to which to connect. If you do not specify **-S**, **langinstall** uses the server specified by your DSQUERY environment variable. If DSQUERY is not set, **langinstall** attempts to connect to a server named SYBASE.

-**I** *interfaces_file*
specifies the name and location of the interfaces file that **langinstall** searches when connecting to Adaptive Server. If you do not specify **-I**, **langinstall** uses the interfaces file in the directory specified by the SYBASE environment variable. If SYBASE is not set, **langinstall** looks for the default *SYBASE* directory.

-**P** *password*
specifies the "sa" account password. If you omit **-P**, **langinstall** prompts for the "sa" account password.

-**R** *release_number*
specifies the release number, in the format *n.n.n.*, to use to upgrade messages in *master..sysmessages*. Use **-R** only in failure conditions, such as if **langinstall** fails or in case of user error or when you think that messages in *sysmessages* are out of date.

language
is the official name of the language to be installed. You must specify a language.

character_set
is the name of Adaptive Server's default character set. *character_set* indicates the directory name of the localization files for the language. The *common.loc* and *server.loc* localization files for an official language reside in the character set directory *$SYBASE/locales/language/character_set*. You must specify a character set.

-v

prints the version number and copyright message for **langinstall** and then exits.

Usage

- The Adaptive Server installation program runs **langinstall** automatically for a new installation as well as for customers who are upgrading from a previous release.
- **langinstall** does the following:
  - Adds the specified language-specific information to *master..syslanguages* using the **sp_addlanguage** stored procedure. If the language already exists, **langinstall** updates the appropriate row in *syslanguages*.
  - Adds to, updates, and deletes error messages as necessary from *master..sysmessages*.
  - Updates *syslanguages.update*, inserting the new release number.
- **langinstall** validates the entries in the localization file sections that it uses. If anything is missing, **langinstall** prints an error message and does not add the language to *syslanguages*.
- **langinstall** compares the version numbers of each localization file it uses, *common.loc* and *server.loc*. If they are not the same, it prints a warning message. *syslanguages.upgrade* is always set according to the version number in *server.loc*.
- The **-R** parameter forces **langinstall** to collect messages from a release previous to the current one. **langinstall** compares the existing messages with the ones to be installed and replaces any that have changed.

  For example, if the current release is 11.5, and the previous release was 10.0, and you think *sysmessages* may not be correct, include the messages from the previous release in the *syslanguages.upgrade* column (10.0 in this case) by specifying **-R 10.0**. **langinstall** then installs all messages from Adaptive Server 10.0.

Permissions

Only the "sa" account can run **langinstall**.

Tables used

master.dbo.syslanguages, master.dbo.sysmessages

See also

System procedures	sp_addlanguage, sp_addlogin, sp_configure, sp_defaultlanguage, sp_droplanguage, sp_helplanguage
Utility	**srvbuild**

optdiag

Description

Displays optimizer statistics or loads updated statistics into system tables.

Syntax

optdiag [binary] [simulate] statistics
{ -i *input_file* |
database[.*owner*[.[*table*[.*column*]]]]
[-o *output_file*] }
[-U *user_name*]
[-P *password*]
[-I *interfaces_file*]
[-S *server*]
[-v]
[-h]
[-s]
[-T *flag_value*]
[-z *language*]
[-J *client_charset*]
[-a *display_charset*]

Parameters

binary

extracts statistics in human-readable form and in binary form. When used with an input file (**-i** *input_file*), loads binary statistics into system tables.

simulate

specifies that **optdiag** display or load simulated statistics. See the *Performance and Tuning Guide*.

-i *input_file*

– specifies the name of the operating system file to use for **optdiag** input. Specifying an input file causes **optdiag** to update optimizer statistics for the table or column by using the values in the specified file (Also called “input mode”).

database

is the name of the database whose statistics you want displayed. In input mode, **optdiag** uses the database name as specified in the file, and does not accept a database name from the command line.

owner

is the name of a table owner.

- In display mode, if you do not specify an owner, but do specify a table name, **optdiag** displays output for all of the owners of a table.
- In input mode, **optdiag** ignores the table owner specified on the command line and uses the value in the input file.

table
is the name of the table to survey for statistics.

- If the command does not include an owner name or a table name, **optdiag** displays statistics for all tables in the database.
- If the command includes an owner name, but no table name, **optdiag** displays all of the tables that belong to the specified owner.
- In input mode, **optdiag** ignores the table name specified on the command line and uses the value from the input file.

column
is the name of the colum to survey.

- If the command does not include a column name, **optdiag** displays all statistics for a table.
- In input mode, **optdiag** ignores the column name on the command line and uses the values from the input file.

-o *output_file*
specifies the name of an operating system file to store the output from **optdiag**. If a file with the same name already exists, **optdiag** overwrites that file without warning.

-U *user_name*
specifies an Adaptive Server login name.

-P *password*
specifies your Adaptive Server password. If you do not specify the **-P** flag, **optdiag** prompts for a password.

-I *interfaces_file*
specifies the name and location of the interfaces file to use when connecting to Adaptive Server.

If you do not use **-I** and specify an interfaces file name, **optdiag** looks for a file named *interfaces* in the directory specified by the SYBASE environment variable. Then, if SYBASE is not set, **optdiag** looks for the file in the default *$SYBASE* directory.

-**S** *server*
specifies the name of the Adaptive Server to which to connect. **optdiag** looks for this name in the *interfaces* file.

- If you use **-S** without specifying a server name, **optdiag** looks for a server named SYBASE.
- When you do not use **-S**, **optdiag** looks for the server that your DSQUERY environment variable specifies.

-**a** *display_charset*
runs **optdiag** from a terminal whose character set differs from that of the machine on which **optdiag** is running.

- Use **-a** in conjunction with **-J** to specify the character set translation (*.xlt*) file required for the conversion.
- Use **-a** without **-J** only if the client character set is the same as the default character set.

Note The ascii_7 character set is compatible with all character sets. If either the Adaptive Server character set or the client character set is set to ascii_7, any 7-bit ASCII character can pass unaltered between client and server. Any other characters produce conversion errors. For more information on character-set conversion, see the *System Administration Guide*.

-**z** *language*
is the official name of an alternate language that the server uses both for date formats and to display **optdiag** prompts and messages. Without the **-z** flag, **optdiag** uses the server's default language.

You can add languages to Adaptive Server either during or after installation, After Adaptive Server installation, use either the **langinstall** utility or the **sp_addlanguage** stored procedure to add a language.

-**J** *client_charset*
specifies the character set to use on the client. A filter converts input between *client_charset* and the Adaptive Server character set.

- By using **-J** *client_charset*, you request that Adaptive Server convert data to and from *client_charset*, the client's character set.
- By using **-J** without a character set name, you specify character set conversion as NULL; no conversion takes place. Use this -**J** alone when the client and server are using the same character set.
- By omitting **-J**, you set the character set to the default set for the platform. A filter converts input between the default set and the Adaptive Server character set. Keep in mind that the default may not necessarily be the character set that the client is using.

  For more information about character sets and their associated flags, see the *System Administration Guide*.

-**v**
displays the version number of and a copyright message for **optdiag** and exits.

-**h**
displays the **optdiag** syntax help.

-**s**
includes system tables in **optdiag** output. By default, only user tables are included.

-**T***flag_value*
sets trace flags for the **optdiag** session. The **optdiag** trace flags are shown in Table 5-13.

Table 5-13: optdiag trace flags

Flag Value	Meaning
1	Do not stop with a warning if the optdiag version of Adaptive Server in use does not match the Adaptive Server version in the input file.
2	Display status message "Next table is *table_name*" when in input mode.
4	Skip consistency checking for step numbers while loading histograms in input mode.
6	Display lines of input file during input mode. This flag has no effect in display mode.

Note **optdiag** works only with single-byte character sets. If your server is using a multibyte character set, **optdiag** displays a warning message and exits.

Examples

```
optdiag statistics pubs2 -Usa -Ppasswd -o pubs2.opt
```

Displays statistics for all user tables in the *pubs2* database and places the output in the *pubs2.opt* file.

```
optdiag statistics pubs2..titles -Usa -Ppasswd
   -o titles.opt
```

Displays statistics for the *titles* table.

```
optdiag statistics pubs2..titles -Usa -Ppasswd
   -o titles.opt -J roman8 -z french
```

Displays statistics using the roman8 character set and row labels and error messages in French.

```
optdiag binary statistics pubs2..titles.price
   -Usa -Ppasswd -o price.opt
```

Displays binary statistics for the *price* column in the *titles* table.

```
optdiag statistics -i price.opt -Usa -Ppasswd
```

Loads edited statistics from the *price.opt* file.

Usage

- For more information about the **optdiag** command and an explanation of the **optdiag** output, see the *Performance and Tuning Guide*.
- For more information on changing statistics using **optdiag**, see the *Performance and Tuning Guide*.
- By default, **optdiag** does not include the system tables when you display statistics for a database. To include the system tables in the output, use the **-s** flag.

- When you use **binary** mode, **optdiag** displays the human-readable values with comment marks (#s) at the beginning of the lines, as shown in this example:

```
Statistics for column:              "price"
Last update of column statistics: Jan 20 1998  7:16PM
Statistics loaded from Optdiag.
  Range cell density:         0x3f8b9cfefece26bf
# Range cell density:         0.0134830400000000
  Total density:              0x3f8b9cfefece26bf
# Total density:              0.0134830400000000
  Range selectivity:          default used (0.33)
# Range selectivity:          default used (0.33)
  In between selectivity:     default used (0.25)
# In between selectivity:     default used (0.25)
```

- When you use **optdiag** with an input file to change statistics, it ignores all characters after the "#" in a line.
- Converting floating-point values may lead to rounding errors when you use files for input.

  When you are loading statistics on the same hardware platform, edit the statistics using the binary values to provide greater precision.

Byte Ordering and Binary *optdiag* Files

- Do not use the **binary** mode option to move statistics between Adaptive Servers on machines that use different byte ordering.
  - On an incompatible architecture server, always comment out binary statistics and load the human-readable statistics.
  - On a compatible architecture server,you can load either binary statistics or human-readable statistics.

Input Mode

- When you use the **-i** *input_file* syntax, **optdiag** reads the file as named and updates statistics in *sysstatistics*.
- **optdiag** input mode changes the configuration parameter **allow update to system tables** by setting the parameter to 1 at the beginning of the session, and then to 0 at the end of the session.
- During histogram input, the process checks the following rules and displays error messages for any violated rules:
  - The step numbers must increase monotonically, unless the command includes the **-T4** trace flag.

- The column values for the steps must increase monotonically.
- The weight for each cell must be between 0.0 and 1.0.
- The total of weights for a column must be close to 1.0.
- The first cell represents null values, and it must be present, even in columns that do not allow null values. There must be only one cell to represent the null value.
- Two adjacent cells must not both use the **<** (less than) operator.

See also

Commands	create index, delete statistics, set, update statistics
System procedures	sp_addlogin, sp_configure, sp_defaultlanguage, sp_droplanguage, sp_flushstats, sp_helplanguage

showserver

Description

Shows the Adaptive Servers and Backup Servers that are currently running on the local machine.

Syntax

showserver

Parameters

None.

Examples

```
showserver
```

```
USER       PID %CPU %MEM   SZ  RSS TT STAT START  TIME COMMAND
user114276  0.0  1.7  712 1000 ?  S    Apr  5514:05 dataserver
-d greensrv.dat -sgreensrv -einstall/greensrv+_errorlog
sybase    1071  0.0  1.4  408  820 ?  S    Mar 28895:38
/usr/local/sybase/bin/dataserver -d/dev/rsd1f -e/install/errorlog
user128493  0.0  0.0 3692    0 ?  IW   Apr  1  0:10 backupserver -SSYB_BACKUP
-e/install/backup.log -Iinterfaces -Mbin/sybmultbuf -Lus_english -Jiso_1
```

Usage

- **showserver** displays process information about Adaptive Server or Backup Server. If no servers are running, only the header appears.

See also

Utilities	**langinstall**
Commands	**dataserver**, **startserver**
Function	host_name

sqlloc

Description

Installs and modifies languages, character sets, and sort order defaults for Adaptive Server using a graphical user interface based on X11/Motif.

Syntax

```
sqlloc
 [-S server]
 [-U user]
 [-P password]
 [-s sybase_dir]
 [-I interfaces_file]
 [-r resource_file]
```

or

```
sqlloc -v
```

Parameters

-**S** *server*
specifies the name of the Adaptive Server to which to connect.

-**U** *user*
specifies a login name. Logins are case sensitive.

-**P** *password*
specifies the "sa" account password.

-**s** *sybase_dir*
specifies the value to use for the SYBASE environment variable.

-**I** *interfaces_file*
specifies the name and location of the interfaces file to search when connecting to Adaptive Server.

-**r** *resource_file*
executes the specified resource file.

-**v**
prints the version number and copyright message for **sqlloc** and then exits.

Usage

- You must set the SYBASE environment variable to the location of the current version of Adaptive Server before you can use **sqlloc**.
- You must set the DISPLAY environment variable before invoking **sqlloc**, unless you are only using the **-v** parameter to display the version number.
- For more information about the **sqlloc** utility program, see *Adaptive Server Enterprise Installation Guide on UNIX Platforms*.

Permissions

You must be a Sybase System Administrator to use **sqlloc.**

See also

Utilities	**langinstall**, **sqllocres**

sqllocres

Description

Installs and modifies languages, character sets, and sort order defaults for Adaptive Server, using a resource file.

Syntax

```
sqllocres
 [-S server]
 [-U user]
 [-P password]
 [-s sybase_dir]
 [-I interfaces_file]
 [-r resource_file]
```

or

```
sqllocres -v
```

Parameters

-**S** *server*
specifies the name of the Adaptive Server to which to connect.

-**U** *user*
specifies a login name.

-**P** *password*
specifies the "sa" account password.

-**s** *sybase_dir*
specifies the value to use for the SYBASE environment variable.

-**I** *interfaces_file*
specifies the name and location of the interfaces file to search when connecting to Adaptive Server.

-**r** *resource_file*
executes the specified resource file.

-**v**
prints the version number and copyright message for **sqllocres** and then exits.

Usage

- You must set the SYBASE environment variable to the location of the current version of Adaptive Server before you can use **sqllocres**.
- For more information about the **sqllocres** utility program, see the *Adaptive Server Enterprise Installation Guide for UNIX Platforms*.

Permissions

You must be a Sybase System Administrator to use the **sqllocres** utility.

See also

Utilities	**langinstall**, **sqlloc**

sqlupgrade

Description

Upgrades your currently installed release of Adaptive Server to the newest release using a graphical user interface based on X11/Motif.

Syntax

```
sqlupgrade
 [-s sybase_dir]
 [-r resource_file]
```

or

```
sqlupgrade -v
```

Parameters

-**s** *sybase_dir*
specifies the value to use for the SYBASE environment variable.

-**r** *resource_file*
executes the specified resource file.

-**v**
prints the version number and copyright message for **sqlupgrade** and then exits.

Usage

- You must set the SYBASE environment variable to the location of the current version of Adaptive Server before you can use **sqlupgrade**.
- You must set the DISPLAY environment variable before invoking **sqlupgrade**, unless you are only using the **-v** parameter to display the version number.
- For more information about the **sqlupgrade** utility program, see the *Adaptive Server Enterprise Installation Guide for UNIX Platforms.*

Permissions

You must be a Sybase System Administrator to use **sqlupgrade.**

See also

Utilities	**sqlupgraderes**

sqlupgraderes

Description

Upgrades your currently installed release of Adaptive Server to the newest release using resource files.

Syntax

```
sqlupgraderes
 [-s sybase_dir]
 [-r resource_file]
```

or

```
sqlupgraderes -v
```

Parameters

-**s** *sybase_dir*
specifies the value to use for the SYBASE environment variable.

-**r** *resource_file*
executes the specified resource file.

-**v**
prints the version number and copyright message for **sqlupgraderes** and then exits.

Usage

- You must set the SYBASE environment variable to the location of the current version of Adaptive Server before you can use **sqlupgraderes**.
- For more information about the **sqlupgraderes** utility program, see the *Adaptive Server Enterprise Installation Guide for UNIX Platforms*.

Permissions

You must be a Sybase System Administrator to use **sqlupgraderes**.

See also

Utilities	**sqlupgrade**

srvbuild

Description

Creates a new Adaptive Server, Backup Server, Monitor Server, or XP Server with default or user-specified values for key configuration attributes using a graphical user interface based on X11/Motif.

Syntax

srvbuild
[-s *sybase_dir*]
[-l *interfaces_file*]
[-r *resource_file*]

or

srvbuild -v

Parameters

-s *sybase_dir*
specifies the value to use for the SYBASE environment variable.

-l *interfaces_file*
specifies the name and location of the interfaces file to search when connecting to Adaptive Server.

-r *resource_file*
executes the specified resource file.

-v
prints the version number and copyright message for **srvbuild** and then exits.

Usage

- You must set the SYBASE environment variable to the location of the current version of Adaptive Server before you can use **srvbuild**.
- You must set the DISPLAY environment variable before invoking **srvbuild**, unless you are only using the **-v** parameter to display the version number.
- For more information about the **srvbuild** utility program, see the *Adaptive Server Enterprise Installation Guide for UNIX Platforms.*

Permissions

You must be a Sybase System Administrator to use **srvbuild**.

See also

Utilities	**srvbuildres**

srvbuildres

Description

Creates, using resource files, a new Adaptive Server, Backup Server, Monitor Server, or XP Server with default or user-specified values for key configuration attributes.

Syntax

```
srvbuildres
 [-s sybase_dir]
 [-I interfaces_file]
 [-r resource_file]
```

or

```
srvbuildres -v
```

Parameters

-**s** *sybase_dir*
specifies the value to use for the SYBASE environment variable.

-**I** *interfaces_file*
specifies the name and location of the interfaces file to search when connecting to Adaptive Server.

-**r** *resource_file*
executes the specified resource file.

-**v**
prints the version number and copyright message for **srvbuildres** and then exits.

Usage

- You must set the SYBASE environment variable to the location of the current version of Adaptive Server before you can use **srvbuildres**.
- For more information about the **srvbuildres** utility program, see the *Adaptive Server Enterprise Installation Guide for UNIX Platforms*.

Permissions

You must be a Sybase System Administrator to use **srvbuildres**.

See also

Utilities	**srvbuild**

startserver

Description

Starts an Adaptive Server and/or a Backup Server.

Syntax

startserver [[-f *runserverfile*] [-m]] ...

Parameters

-**f** *runserverfile*

specifies the relative path name of a runserver file, which is used as a reference each time you start an Adaptive Server or Backup Server. By default, the runserver file is in the current directory and is named *RUN_servername*. If you start a second Adaptive Server on the same machine, **startserver** creates a new runserver file named *RUN_servername*.

-**m**

starts Adaptive Server in single-user mode, allowing only one System Administrator to log in, and turns the **allow updates to system tables** configuration parameter **on**. Use this mode to restore the *master* database. The System Administrator can use the **dbo use only** parameter of **sp_dboption** for system administration activities that require more than one process, such as bulk copying or using the data dictionary. **startserver** normally starts up only one server per node.

The **-m** parameter creates an *m_RUNSERVER* file and overwrites any existing *m_RUNSERVER* file.

Examples

```
startserver
```

Starts an Adaptive Server named SYBASE from the runserver file named *RUN_servername* in the current directory.

```
startserver -f RUN_MYSERVER -f RUN_SYB_BACKUP
```

Starts an Adaptive Server named MYSERVER and a Backup Server named SYB_BACKUP.

```
startserver -f RUN_SYB_BACKUP
```

Starts only the Backup Server SYB_BACKUP.

Usage

- **startserver** uses the information in the runserver file to start an Adaptive Server or Backup Server. The master device must be writable by the user who starts Adaptive Server.

The **startserver** command creates the Adaptive Server error log file (named *errorlog*) in the directory where the server is started, and adds this information as part of the **-e** parameter in the Adaptive Server executable line in the runserver file. If a second Adaptive Server is started on the same machine, a new error log named *errorlog_servername* is created; this information is added to that server's runserver file. The user must have execute permission on the specified runserver file.

- You can start multiple servers by specifying more than one runserver file, as shown in example 2. You can specify **-m** after each **-f** *runserverfile*.
- Adaptive Server derives its running environment from values in the *config* file. Run the system procedure **sp_configure** or edit the *config* file to see or change configuration parameters.
- To ensure the integrity of your Adaptive Server, it is important that you apply appropriate operating system protections to the **startserver** executable and the runserver file.

The Runserver File •

The runserver file, which is created by **srvbuild** during installation, contains the **dataserver** command to start Adaptive Server or the **backupserver** command to start Backup Server. By default, the runserver file is in the current directory and is named *RUN_servername*. You can edit the runserver file to correct the options and parameters for the commands. The following example shows two sample runserver files:

Runserver file for server MYSERVER:

```
#!/bin/sh
#
# Adaptive Server Information:
#  name:                          /MYSERVER
#  master device:                 /remote/Masters/myserver_dat
#  master device size:            10752
#  errorlog:                      /remote/serverdev/install/errorlog
#  interfaces:                    /remote/serverdev/interfaces
#
#
/remote/serverdev/bin/dataserver -d/remote/Masters/myserver_dat \
-sMYSERVER -e/remote/serverdev/install/MYSERVER_errorlog \
-i/remote/serverdev &
```

Runserver file for backup server SYB_BACKUP:

```
#!/bin/sh
#
# Backup Server Information:
```

```
#  name:                           SYB_BACKUP
#  errorlog:                       /remote/serverdev/install/backup.log
#  interfaces:                     /remote/serverdev/interfaces
#  location of multibuf:           /remote/serverdev/bin/sybmultbuf
#  language:                       us_english
#  character set:                  iso_1
#  tape configuration file:        /remote/serverdev/backup_tape.cfg
#
#
/remote/serverdev/bin/backupserver -SSYB_BACKUP \
-e/remote/serverdev/install/backup.log \
-I/remote/serverdev/interfaces \
-M/remote/serverdev/bin/sybmultbuf -Lus_english -Jiso_1 \
-c/remote/serverdev/backup_tape.cfg
```

See also

Command	disk mirror, disk remirror, disk unmirror
Utilities	**backupserver**, **dataserver**

sybload

Description

Uploads Sybase products from the distribution media and builds the Sybase installation directory.

Syntax

sybload [-D]

Parameters

-D

uploads Adaptive Server from the distribution media.

Usage

- **sybload** is a command-line utility.
- You must set the SYBASE environment variable to the location of the current version of Adaptive Server before you can use **sybload**.
- **sybload** issues **rsh** commands to the UNIX shell to unload products remotely. Be sure the default version of the program on the host system is **/bin/rsh,** or **sybload** will fail.
- For more information about installing Adaptive Server, see the Sybase installation documentation for your platform and the administration documentation for your operating system.
- You cannot perform a remote installation from CD-ROM using **sybload**. For information on how to install remotely from a CD-ROM, see the installation documentation for your platform.
- **sybload** prompts you for installation information. Table 5-14 summarizes the information you need to enter.

Table 5-14: Summary of sybload prompts for tape installation

Prompt	Value
Sybase directory	Confirm that the current directory is the Sybase installation directory, or specify the correct directory path.
Local or remote installation	Enter either: "L" – local "R" – remote installation
Name of the non-rewinding tape drive	Enter the name of the device from which you are loading your SYBASE software—*must be the non-rewinding name*.
Customer Authorization String (CAS)	Enter the string from the software packaging that allows you to access your products.
Sybase products	Select products to install from the sybload menu: -Enter the number of each product you plan to install -Press Return after each number -Press Return twice to enter a blank line when finished

Prompt	Value
Product confirmation	The sybload utility lists the products you have chosen. Enter: "y" – correct "q" – quit *Any_other_character* to display the menu and choose again

- For more information about **sybload**, see the *Adaptive Server Enterprise Installation Guide for UNIX Platforms*.

sybsetup

Description

Installs and configures Adaptive Server from a single location using a graphical user interface based on X11/Motif.

Syntax

sybsetup
[-c *devicename*]
[-t *devicename*]
[-l *local_source*]
[-r *remote_source*]

Parameters

-**c** *devicename*
is the CD-ROM directory path of your Adaptive Server distribution media.

-**t** *devicename*
is the tape device directory path of your Adaptive Server distribution media.

-**l** *local_source*
is the path name of the local tape device.

-**r** *remote_source*
is the path name of the remote tape device.

Usage

- Before you can use **sybsetup** you must:
  - Set the SYBASE environment variable to the location of the current version of Adaptive Server .
  - Set the DISPLAY environment variable
- For more information about the **sybsetup** utility program, see the *Adaptive Server Enterprise Installation Guide for UNIX Platforms*.

Permissions

You must be the Sybase System Administrator to use **sybsetup**.

Index

Symbols

!! (exclamation points) operating system commands prefix (isql) 156
\ (backslash)
  escaping special characters 80
\ (backslash) data field terminator in interactive bcp 36
\0 (null) character terminator in interactive bcp 36
\gt (redirect out) in isql 9
\isql 9

A

Adaptive Server
  configuring (sybsetup) 187
  executing (dataserver) 126
  installing (sybsetup) 187
  rolling back processes 49
  upgrading (sqlupgrade) 178
  upgrading with resource files (sqlupgraderes) 179
Adding
  network transport addresses in dsedit 61
  server entries in dscp 71
  server entries in dsedit 61
allow_dup_row option to create index, and bcp 53
Application programs, copying data from 41
ASCII format, bcp and 29
ascii_7 character set
  bcp compatibility 95
  defncopy and 130

B

Backing up, compared to bulk copying 56
Backslash (\›
  escaping special characters 80
  terminator in interactive bcp 36
Backup Server 85
  See also backupserver utility command 85
backupserver utility command 85
  character set, default 86
  defined 85
  error log file 85
  error messages in 89
  full path names, specifying 86
  interfaces file 89
  interfaces file, name and location of 85
  language, default 86
  LC_ALL environment variable 86
  permissions required for 89
  server connections, number of 85
  server names, specifying 85
  starting servers and 88
  trace flags 89
-bbatch_size parameter 49
bcp utility command 56, 90, 104
  allow_dup_row option to create index 53
  ASCII format and 29
  ascii_7 character set compatibility 95
  batch operations in 49
  batch size settings 93
  -bbatch_size parameter 49
  binary format and 13
  character format 29
  character format, default 29
  character formats accepted 13
  character set defaults 94
  configuration parameters 52
  copying data in 19, 21, 44, 47
  copying data out 41, 43, 51
  copying in batches of table rows 49
  data integrity 55
  data storage size in 102
  data transfer, preparing for 14
  datatypes 104
  default values for data 55

defaults for columns and datatypes 55
defined 90
described 12
drop index, using 100
drop trigger, using 100
dump database and 56, 100
dump transaction and 100
error files 53
errors in data conversion 53
examples for using 97
fast version of 16, 99
fast version of, and data recoverability 17
field lengths 29
field terminators 30, 102
file storage types in 102
format files 37, 40
IDENTITY columns and 26
improving performance of operations 52
improving recoverability when copying data in 49
index creation 53
indexes and triggers, dropping 18
insert and 56
interactive mode 31, 36
languages, using alternate 48
load database and 56
load transaction and 56
maximum speed, enabling 55
native format option and 29
non-character to character datatype default field lengths 34
non-interactive 29
non-iso_1 data files and 48
operating system differences 102
other Adaptive Server utilities and 56
page allocations, increasing for 21
partitioned tables and 19, 27, 50
password encryption in 92
performance issues with 17, 26, 55
performance speed in 100
permissions required for 14, 104
prefix length 102
prompts and responses. See Interactive bcp 30, 31
prompts in 100, 102
responses in 100
rolling back processes 49
row terminators 30
rules and copying data 55
select into/bulkcopy/pllsort and 18, 100
slow version and deadlocks on index pages 21
slow version of 16, 99
sp_dboption 18, 100
space needed for data 18
special characters, handling 96
speed, and indexes and triggers 17
speed, modes of 16
storage types 104
system data format (SDF) output in 41
table defaults and copying data 55
terminators used in 13
transferring data between programs 12
triggers and data copying 55
triggers not fired on target table 17
unique IDENTITY column values and 26
warning against using native format 92
Binary data
bcp and 13
default in interactive bcp 33
Buffer, query 4
Building master devices 122
See also buildmaster utility command 122
buildmaster utility command 122
defined 122
examples 123
master database and 122
model database and 122
permissions required for 123
Bulk Copy Process 56, 90, 104
See also bcp utility command 90

C

Carriage-return data field terminator (\\r) in interactive bcp 36
char datatype, and bcp 29
Character formats
bcp 13
default in bcp 29
terminators for 36
Character sets
backupserver, default in 86
bcp, defaults in 94

converting from non-character data 34
installing and modifying (sqllocres) 177
installing and modifying, in GUI (sqlloc) 176
loading, with charset 125
platform default 48
charset utility command 125
defined 125
permissions required for 125
settings for 125
Closing a session
dscp 69
dsedit 59
Column precision in numeric or decimal storage formats 40
Column scale in numeric or decimal storage formats 40
Columns
datatype sizes and 34
default values and bcp 55
fixed- and variable-length 33
null 53
separator character (isql) 3
Comma-delimited output 41, 43
Command terminator (isql) 3
statistics option interaction 7
Configuring
Adaptive Server (sybsetup) 187
Conventions, syntax xiii
Copying
between sessions in dscp 73
different sessions in dscp 74
new entries with dscp 73
server entries with dscp 73
server entries with dsedit 61
Copying data in
batch operations in 49
improving recoverability after rolling back 49
parallel bcp, requirements for 22, 26
partitioned tables 20
partitions, random use of 20
steps, using fast version of bcp 19
Copying data in with interactive bcp 44, 47
compatibility of datatypes, and failure 46
delimiters 46
error files and 54
field lengths 44
Copying data out with interactive bcp 41, 43
delimiters 42
error files and 54
fixed-length fields 41
for other software 41
text and image data 51
Copying definitions 129, 134
create index command, bcp and duplicate rows 53
Creating
new servers (srvbuild) 180
new servers using resource files (srvbuildres) 181
tempdb databases with buildmaster 122

D

Data
changing with Adaptive Server commands 56
conversion errors 53, 54
importing and exporting with bcp 12
moving with Adaptive Server commands 56
padding with spaces in interactive bcp 33
parsing. See Field terminators 30
permission required to copy into tables 14
recoverability 17
transferring from different programs using bcp 12
Database management systems, other 41
Database objects
copying using bcp 90, 104
Databases, copying with bcp 56
dataserver utility command 126
defined 126
passwords, generating new 127
permissions required for 128
Datatypes
bcp file storage types for 31, 32
bcp format files for 37
bcp, used in 104
char 29
copying and compatibility 46
default values and bcp 55
field lengths in interactive bcp 31, 36
implicit conversions of 31, 32
non-character to character default field lengths in bcp 34
storage (SYB) 38

default network packet size configuration parameter 52
Defaults
bcp data conversion 34
bcp prompts 31, 36
character sets in backupserver 86
character sets, in bcp 94
copying into tables using data 55
copying, with defncopy 129
languages in backupserver 86
prompts in interactive bcp 31, 36
select into/bulkcopy/pllsort option settings in new databases 18
Definitions
backupserver utility command 85
bcp utility command 90
buildmaster utility command 122
charset utility command 125
copying. See defncopy utility command 129
dataserver utility command 126
defncopy utility command 129
dscp utility command 139
dsedit utility command 141
isql utility command 150
optdiag utility command 168
showserver utility command 175
sqlloc utility command 176
sqllocres utility command 177
sqlupgrade utility command 178
sqlupgraderes utility command 179
srvbuild utility command 180
srvbuildres utility command 181
startserver utility command 182
sybload utility command 185
sybsetup utility command 187
defncopy utility command 129, 134
ascii_7 character set and 130
defined 129
encrypted text 132
examples 131
failure due to long comments 133
passwords and crashing 129
permissions required for 133
Report Workbench, incompatibility with 129
Deleting
server entries (dscp) 75
Delimiters
copying data in with 46
copying data out with 42
DISPLAY environment variable
setting, in dsedit 58
drop index command, bcp and 100
drop trigger command, bcp and 100
dscp utility command 139
commands, table of 76
defined 67, 139
entries, creating new, by copying 73
examples of 139
exit command 76
help 68, 77
permissions required for 139
quit command 76
server attributes 70
server entries, adding 71
server entries, copying 73
server entries, deleting 75
server entries, listing 74
server entries, modifying 72
server entry contents, viewing 74
sessions, closing 69
sessions, copying between 73
sessions, copying to different 74
sessions, listing 68
sessions, opening 68
starting 68
dsedit utility command 141
defined 141
described 58
Directory Service Session screen 59
DISPLAY environment variable and 58
does not start 64
network transport addresses, adding 61
network transport addresses, editing 62
permissions required for 58, 141
remote machines, running from 58
Select a Directory Service screen 58
server entries, adding 61
server entries, cannot add, modify, or delete 64
server entries, copying 61
server entries, modifying 61
sessions, closing 59
sessions, opening 59
starting 58

troubleshooting 57, 64
xd2 Unable to open X displayxd3 error message 64
DSLISTEN environment variable
backupserver and 88
dump database command
bcp and 56
fast version of bcp and 100
dump transaction command
bcp and 56
error message recommending use of dump database 18
fast version of bcp and 100
isql utility command 154
optdiag utility command 172
showserver utility command 175
startserver utility command 182, 183
Exclamation points (!!) operating system commands prefix (isql) 156
Executing Adaptive Server 126
See also dataserver utility command 126
exit command
in dscp 76
in isql 2
Exporting data. See Copying data out with interactive bcp 41

E

Echo input (isql) 9
Editing
interfaces files 58, 59
network transport addresses in dsedit 62
Encrypted hidden text 132
Environment variables
DSLISTEN (backupserver) 88
LANG (backupserver) 86
LC_ALL (backupserver) 86
Error files
bcp and 53, 54
copying out and 54
Error log files
for backupserver 85
Error messages
dsedit and X display 64
dump database use over dump transaction 18
in backupserver 89
select into/bulkcopy/pllsort in fast bcp on tables with no indexes or triggers 18
Errors
in data conversions 53, 54
operations, saving during copy in 53
operations, saving during copy out 54
Examples
bcp utility command 97
buildmaster utility command 123
defncopy utility command 131
dscp utility command 139

F

Fast version of bcp 16, 99
copying data in 19
data recoverability and 17
Field lengths
copy in 44
in interactive bcp 33, 35
prefix length, integer value in 39
Field terminators 30
bcp 102
interactive bcp 35, 36
Files
See also Format files in bcp 37
error (bcp) 53
Fixed-length fields 41
Font conventions xiii
Format files in bcp 37, 40
column numbers 37
column precision 40
column scale 40
elements used in 37
format used for 37
host file column order 38
host file data length 39
host file datatype storage formats (SYB) 38
number of records in 37
prefix length, integer value in 39
saving 37
server column names 40
server column orders 40

terminators 40
version number in 37
Full path names, specifying
in Backup Server 86

H

help command (dscp) 68, 77
Hidden encrypted text 132
Host file
isql and reading 9
Host files
column order 38
data length 39
datatype storage formats (SYB) 38
interactive bcp native format 33

I

IDENTITY columns
bcp and 26
parallel bcp and 26
ignore_dup_key option, create index, and bcp 53
image datatype
interactive bcp and 33
Importing data. See Copying data in with interactive bcp 44
Indexes
bcp, dropping before using 18
slowing down bcp 17
insert command
bulk copying, comparing 56
Installing
Adaptive Server (sybsetup) 187
character sets (sqllocres) 177
character sets, in GUI (sqlloc) 176
languages (sqllocres) 177
languages, in GUI (sqlloc) 176
languages, new (langinstall) 166, 167
sort orders (sqllocres) 177
sort orders, in GUI (sqlloc) 176
installjava (UNIX) utility command($startrange> 143, 146
Interactive bcp 31, 36
backslash terminator (\› 36
binary data, default 33
carriage-return data field terminator (\\r) 36
character format files, terminators for 36
copying data for other software 31
copying data in 44, 47
copying data out 41, 43
defaults for prompts 31, 36
field length 33, 35
field terminators 35, 36
fields, prefix length of 32, 33
file storage types 31, 32
image datatype, default for 33
implicit conversion of datatypes 31, 32
null or invisible character terminator (\0) 36
null values and 33
padding data with spaces 33
row terminators 35, 36
storage length 33, 35
tab data field terminator (\) 36
terminator for new lines (\\n) 36
terminators for tabular data preparation 36
terminators, field and row 35, 36
text data copying,default for 33
Interactive SQL parser 150
See also isql utility command 150
Interfaces files
backupserver and 89
backupserver, specifying names and locations of 85
dscp sessions, opening 68
dscp, viewing and editing with 139
dsedit, editing with 59
dsedit, viewing and editing in GUI with 141
opening for editing 58
iso_1 character set 48
isql utility command 150
\< (redirect in) symbol> 9
\gt (redirect out) symbol 9
column-separator character 3
command terminator 3
command terminator, changing 6
command terminator, resetting 152
correcting typing errors 4
defined 150
echo input 9

examples of 154
exit command 2
host files, reading 9
line numbers, removing 3, 9
maximum statement size 3
network packet size, setting 8
network packet size, specifying in 8
packet size, setting 8
queries, editing 4
query buffer, resetting 4
quit command in 2
reset command in 4
statistics 7
statistics option with command terminator 7
Transact-SQL, using with 2

J

Java-SQL utilities
installjava (UNIX) 143, 146

L

LANG environment variable in backupserver 86
langinstall utility command 166, 167
defined 166
permissions required for 167
Languages
alternate, with bcp 48
installing and modifying (sqllocres) 177
installing and modifying, in GUI (sqlloc) 176
installing new (langinstall) 166, 167
LC_ALL environment variable in backupserver 86
Line numbers, removing in isql 3, 9
load database command and bcp 56
load transaction command
bcp and 56
Loading character sets 125
See also charset utility command 125
Locations of backupserver error log files 85

M

master databases
buildmaster, creating with 122
Master devices, building 122
See also buildmaster utility command 122
max network packet size configuration parameter 52
model databases
creating with buildmaster 122
Modifying
character sets (sqllocres) 177
character sets, in GUI (sqlloc) 176
languages (sqllocres) 177
languages, in GUI (sqlloc) 176
server entries (dscp) 72
server entries with dsedit 61
sort orders (sqllocres) 177
sort orders, in GUI (sqlloc) 176

N

Network packet size
isql, specifying in 8
Network transport addresses
adding with dsedit 61
dsedit, editing with 62
Network transports 59
New servers, creating 180
See also srvbuild utility command 180
New servers, creating, using resource files 181
See also srvbuildres utility command 181
New-line terminator (\\n)
in bcp 29
in interactive bcp 36
Noncharacter datatypes, operating system format for 32
Non-printable characters, host file 33
Notes
ascii_7 character set compatibility 130
ascii_7 character set compatibility in bcp 95
batch size settings in bcp 93
bcp and performance speed 100
copying in data to a view, usingbcp 12
defncopy and Report Workbench 129
hidden encrypted text 132
null terminator versus no terminator 36

operating system differences in bcpbcp utility command 102
passwords, using 80
security 80
select into/bulkcopy/pllsort and copying out data in bcp 18
slow version of bcp and deadlocks 21
special characters, handling in bcp 96
triggers not fired by bcp in target table 17
Null character terminator (\0) in interactive bcp 36
Null values 53
interactive bcp and 33
Number (quantity of)
go command executions 156
server connections to backupserver 85
Numbers
line, removing from isql 3, 9
Numeric datatypes
bcp conversion to character storage 34
column precision 40
column scale 40
operating system format for 29

O

Opening
sessions in dscp 68
sessions in dsedit 59
Operating system files
permission required for copying tables into 14
Operating systems
commands prefix (!!) (isql) 156
non-character datatype formatting 32
numeric datatype formatting 29
optdiag utility command 168
defined 168
examples of 172
permissions required for 174
Output formats, data. See Copying data out with interactive bcp 41

P

Packet size, network
bcp, specifying with 51
isql, specifying in 8
specifying in isql 8
Padding data and bcp 35
Parallel bcp
copying to a specific partition 22, 25
different methods of using 24, 26
IDENTITY columns and 26
syntax 25
Partitioned tables in bcp 19, 27
copying data into randomly 20
copying data into, methods for 20
page allocations, increasing 21
Passwords
bcp encryption 92
defncopy and 129
new, generating 127
notes about using 80
Performance
bcp issues and 17, 26, 55
bulk copy and packet size 52
isql network packet size and 8
Permissions required for
backupserver utility command 89
bcp utility command 14, 104
buildmaster utility command 123
charset utility command 125
dataserver utility command 128
defncopy utility command 133
dscp utility command 139
dsedit utility command 141
dsedit, required for 58
langinstall utility command 167
operating system file, copying tables into 14
optdiag utility command 174
sqlloc utility command 176
sqllocres utility command 177
sqlupgrade utility command 178
sqlupgraderes utility command 179
srvbuild utility command 180
srvbuildres utility command 181
sybload utility command 186
sybsetup utility command 187
tables, copying data into 14
Prefix field lengths in interactive bcp 32, 33
Prefix length 39

Prefix lengths in bcp 102
Prompts
bcp. See Interactive bcp 31
in bcp 100
Prompts in bcp utility command 102

Q

Query buffer, resetting 4
Quick reference
dscp commands 76
quit command
in dscp 76
in isql 2
Quotation marks (xd2 xd3) for enclosing special characters 80

R

Redirect in symbol (\isql 9
Redirect out symbol (\gt) in isql 9
Remote machines 58
Remote machines, using dsedit utility on 58
Report Workbench, incompatibility with defncopy 129
reset command (isql) 4
Responses in bcp 100
Roll-back processes 49
Row terminators
bcp, in 30
interactive bcp, in 35, 36
Rows in bcp, and erroneous table 54
Rows, table
bulk copying and failed 49
length 24
Rules
defncopy, copying with 129
tables, copying data into 55

S

Saving
errors during copy in operations 53
errors during copy out operations 54
format files (bcp) 37
Security Mechanism server attribute (dscp) 71
Security, notes about 80
Select a Directory Service screen 58
select into command, compared to bulk copying 56
select into/bulkcopy/pllsort database option
bcp and 18, 100
Server column
names 40
orders 40
Server connections
backupserver, number of 85
Server entries
contents, viewing with dscp 74
copying, with dsedit 61
dscp, listing with 74
modifying, with dsedit 61
server entries, adding 61
Server Entry Editor window 59
Server Name server attribute (dscp) 70
Server Object Version server attribute (dscp) 70
Server Service server attribute (dscp) 70
Server Status server attribute (dscp) 70
Servers
backupserver, specifying names in 85
backupserver, starting in 88
modes of speed when using bcp 16
startserver utility command 182
startserver, starting in 182
Sessions
listing (dscp) 68
Sever attributes (dscp) 70
Show servers 175
See also showserver utility command 175
showserver utility command 175
defined 175
examples of 175
Size
data storage in bcp 102
packet size 51
text or image data 51
Slow version of bcp 16, 99
deadlocks on index pages 21
Sort orders
installing and modifying (sqllocres) 177

installing and modifying, in GUI (sqlloc) 176
sp_dboption system procedure and bcp 18, 100
Space requirements and bcp steps 18
Special characters
bcp, handling in 96
utility commands, use in 80
Specifying full path names in Backup Server 86
Specifying server names in backupserver 85
SPX/IPX addresses for interfaces entries 62
SQL parser utility. See isql utility command 150
sqlloc utility command 176
defined 176
permissions required for 176
sqllocres utility command 177
defined 177
permissions required for 177
sqlupgrade utility command 178
defined 178
permissions required for 178
sqlupgraderes utility command 179
defined 179
permissions required for 179
srvbuild utility command 180
defined 180
permissions required for 180
srvbuildres utility command 181
defined 181
permissions required for 181
Starting
dscp utility 68
dsedit utility command 58
servers (startserver) 182
startserver utility command 182
backupserver and 88
defined 182
examples of 182
runserver file examples 183
Statistics
displaying (optdiag) 168
isql 7
loading updated (optdiag) 168
Storage types used in bcp 104
Stored procedures
copying with defncopy 129
sybload utility command 185
defined 185
permissions required for 186
sybmultbuf, using to start Backup Server 86
sybsetup utility command 187
defined 187
permissions required for 187
Symbols, field terminator (bcp) 36
Syntax conventions xiii
System data format (SDF) output in bcp 41
System procedure (sp_dboption) 18

T

Tab data field terminator (\) in interactive bcp 36
Table rows
copying in batches 49
length 24
Tables
bcp character set defaults 94
bcp prompts in 102
permission required for copying data into 14
permission required to copy into operating system files 14
utility commands 80
Tabular data, copying 36
Tabular output 41, 43
TCP/IP addresses for interfaces entries 62
tempdb databases, buildmaster creation of 122
Terminators (bcp) 40
changing 30
defined 13
field and row, and 35, 36
other programs, using for 36
text datatype
interactive bcp and 33
Trace flags and backupserver 89
Transact-SQL
using with isql 2
Transferring data from programs using bcp 12
Transport Address server attribute (dscp) 71
Transport Type server attribute (dscp) 70
Triggers
bcp, dropping before using 18
copying with defncopy 129
slowing down bcp 17
tables, copying data into 55

Troubleshooting dsedit 64

U

Unlogged transactions 17
Upgrading Adaptive Server 178
  See also sqlupgrade utility command 178
Upgrading Adaptive Server using resource files 179
  See also sqlupgraderes utility command 179
Uploading Sybase products 185
  See also sybload utility command 185
Utilities, other Adaptive Server, and bcp 56
Utility commands
  special characters, using 80
  table of commands 80

V

Views, copying, with defncopy 129

W

Warnings
  bcp, using native format of 92
  defncopy failure due to long comments 133

X

xd2 xd3 (enclosing special characters) 80