SYBASE®

Utility Guide

Sybase IQ

15.1

DOCUMENT ID: DC00168-01-1510-02

LAST REVISED: December 2009

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

About This Book

Subject

This book provides reference material for the utility programs used with Sybase® IQ. Utility programs are commands that you invoke directly from the operating system. Other books provide more context on how to carry out particular tasks. This reference book is the place to look for information such as available syntax, parameters, and options.

Audience

This book is a reference for all users of Sybase IQ.

Related documents

The Sybase IQ 15.1 documentation set includes:

- *Release Bulletin* provides information about last-minute changes to the product and documentation.
- *Installation and Configuration Guide* provides platform-specific instructions on installing, migrating to a new version, and configuring Sybase IQ for a particular platform.
- *Advanced Security in Sybase IQ* covers the use of user encrypted columns within the Sybase IQ data repository. You need a separate license to install this product option.
- *Error Messages* lists Sybase IQ error messages referenced by Sybase error code, SQLCode, and SQLState, and SQL preprocessor errors and warnings.
- *IMSL Numerical Library User's Guide: Volume 2 of 2 C Stat Library* contains a concise description of the IMSL C Stat Library time series C functions. This book is only available to RAP - The Trading Edition™ Enterprise users.
- *Introduction to Sybase IQ* includes hands-on exercises for those unfamiliar with Sybase IQ or with the Sybase Central™ database management tool.
- *Large Objects Management in Sybase IQ* explains storage and retrieval of Binary Large Objects (BLOBs) and Character Large Objects (CLOBs) within the Sybase IQ data repository. You need a separate license to install this product option.
- *New Features in Sybase IQ 15.0* documents new features and behavior changes for version 15.0.

- *New Features Summary Sybase IQ 15.1* summarizes new features and behavior changes for the current version.
- *Performance and Tuning Guide* describes query optimization, design, and tuning issues for very large databases.
- *Quick Start* lists steps to build and query the demo database provided with Sybase IQ for validating the Sybase IQ software installation. Includes information on converting the demo database to multiplex.
- *Reference Manual* – Includes two reference guides to Sybase IQ:
  - *Reference: Building Blocks, Tables, and Procedures* describes SQL, stored procedures, data types, and system tables that Sybase IQ supports.
  - *Reference: Statements and Options* describes the SQL statements and options that Sybase IQ supports.
- *System Administration Guide* – Includes two volumes:
  - *System Administration Guide: Volume 1* describes startup, connections, database creation, population and indexing, versioning, collations, system backup and recovery, troubleshooting, and database repair.
  - *System Administration Guide: Volume 2* describes writing and running procedures and batches, programming with OLAP, accessing remote data, setting up IQ as an Open Server, scheduling and event handling, programming with XML, and debugging.
- *User-Defined Functions Guide* provides information about the user-defined functions, their parameters, and possible usage scenarios.
- *Using Sybase IQ Multiplex* tells how to use multiplex capability, designed to manage large query loads across multiple nodes.

Sybase IQ and SQL Anywhere

Because Sybase IQ is an extension of SQL Anywhere Server, a component of the SQL Anywhere® package, Sybase IQ supports many of the same features as SQL Anywhere Server. The IQ documentation set refers you to SQL Anywhere documentation, where appropriate.

Documentation for SQL Anywhere includes:

- *SQL Anywhere Server – Database Administration* describes how to run, manage, and configure SQL Anywhere databases. It describes database connections, the database server, database files, backup procedures, security, high availability, and replication with Replication Server®, as well as administration utilities and options.
- *SQL Anywhere Server – Programming* describes how to build and deploy database applications using the C, C++, Java, PHP, Perl, Python, and .NET programming languages such as Visual Basic and Visual C#. This book also describes a variety of programming interfaces such as ADO.NET and ODBC.
- *SQL Anywhere Server – SQL Reference* provides reference information for system procedures, and the catalog (system tables and views). It also provides an explanation of the SQL Anywhere implementation of the SQL language (search conditions, syntax, data types, and functions).
- *SQL Anywhere Server – SQL Usage* describes how to design and create databases; how to import, export, and modify data; how to retrieve data; and how to build stored procedures and triggers.

You can also refer to the SQL Anywhere documentation in the SQL Anywhere 11.01 collection at Product Manuals at http://www.sybase.com/support/manuals/.

Documentation for Sybase Software Asset Management (SySAM) includes:

- *Sybase Software Asset Management (SySAM) 2* introduces asset management concepts and provides instructions for establishing and administering SySAM 2 licenses.
- *SySAM 2 Quick Start Guide* tells you how to get your SySAM-enabled Sybase product up and running.
- *FLEXnet Licensing End User Guide* explains FLEXnet Licensing for administrators and end users and describes how to use the tools that are part of the standard FLEXnet Licensing distribution kit from Sybase.

Other sources of information

Use the Sybase Getting Started CD, the SyBooks™ CD, and the Sybase Product Manuals Web site to learn more about your product:

- The Getting Started CD contains release bulletins and installation guides in PDF format, and may also contain other documents or updated information not included on the SyBooks CD. It is included with your software. To read or print documents on the Getting Started CD, you need Adobe Acrobat Reader, which you can download at no charge from the Adobe Web site using a link provided on the CD.

- The SyBooks CD contains product manuals and is included with your software. The Eclipse-based SyBooks browser allows you to access the manuals in an easy-to-use, HTML-based format.

  Some documentation may be provided in PDF format, which you can access through the PDF directory on the SyBooks CD. To read or print the PDF files, you need Adobe Acrobat Reader.

  Refer to the *SyBooks Installation Guide* on the Getting Started CD, or the *README.txt* file on the SyBooks CD for instructions on installing and starting SyBooks.

- The Sybase Product Manuals Web site is an online version of the SyBooks CD that you can access using a standard Web browser. In addition to product manuals, you will find links to EBFs/Maintenance, Technical Documents, Case Management, Solved Cases, newsgroups, and the Sybase Developer Network.

  To access the Sybase 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.

❖ Finding the latest information on product certifications

1. Point your Web browser to Technical Documents at http://certification.sybase.com/ucr/search.do.
2. Either select the product family and product under Search by Base Product; or select the platform and product under Search by Platform.
3. Select Search to display the availability and certification report for the selection.

❖ Finding the latest information on component certifications

1. Point your Web browser to Availability and Certification Reports at http://certification.sybase.com/.
2. Either select the product family and product under Search by Base Product; or select the platform and product under Search by Platform.
3. Select Search to display the availability and certification report for the selection.

❖ Creating a personalized view of the Sybase Web site (including support pages)

Set up a MySybase profile. MySybase is a free service that allows you to create a personalized view of Sybase Web pages.

1. Point your Web browser to Technical Documents at http://www.sybase.com/support/techdocs/.
2. Click MySybase and create a MySybase profile.

Sybase EBFs and software maintenance

❖ Finding the latest information on EBFs and software maintenance

1. Point your Web browser to the Sybase Support Page at http://www.sybase.com/support.
2. Select EBFs/Maintenance. If prompted, enter your MySybase user name and password.
3. Select a product.
4. Specify a time frame and click Go. A list of EBF/Maintenance releases is displayed.

   Padlock icons indicate that you do not have download authorization for certain EBF/Maintenance releases because you are not registered as a Technical Support Contact. If you have not registered, but have valid information provided by your Sybase representative or through your support contract, click Edit Roles to add the “Technical Support Contact” role to your MySybase profile.
5. Click the Info icon to display the EBF/Maintenance report, or click the product description to download the software.

Syntax conventions

This documentation uses these conventions in syntax descriptions:

- **Keywords** SQL keywords are shown in UPPERCASE. However, SQL keywords are case-insensitive, so you can enter keywords in any case; SELECT, Select, and select are equivalent.
- **Placeholders** Items that must be replaced with appropriate identifiers or expressions are shown in *italics*.
- **Continuation** Lines beginning with an ellipsis (...) are a continuation of the statements from the previous line.

- **Repeating items** Lists of repeating items are shown with an element of the list followed by an ellipsis (...). One or more list elements are allowed. If multiple elements are specified, they must be separated by commas.
- **Optional portions** Optional portions of a statement are enclosed by square brackets. For example:

  ```
  RELEASE SAVEPOINT [ savepoint-name ]
  ```

  The square brackets indicate that the *savepoint-name* is optional. Do not type the brackets.
- **Options** When none or only one of a list of items must be chosen, the items are separated by vertical bars and the list enclosed in square brackets. For example:

  ```
  [ ASC | DESC ]
  ```

  The square brackets indicate that you can choose ASC, DESC, or neither. Do not type the brackets.
- **Alternatives** When precisely one of the options must be chosen, the alternatives are enclosed in curly braces. For example:

  ```
  QUOTES { ON | OFF }
  ```

  The curly braces indicate that you must include either ON or OFF. Do not type the brackets.

Typographic conventions

Table 1 lists the typographic conventions used in this documentation.

Table 1: Typographic conventions

Item	Description
`Code`	SQL and program code appears in a monospaced (fixed-width) font.
`User entry`	Text entered by the user is shown in a monospaced (fixed-width) font.
emphasis	Emphasized words are shown in italic.
file names	File names are shown in italic.
database objects	Names of database objects, such as tables and procedures, are shown in bold, sans serif type in print, and in italic online.

The demo database

Sybase IQ includes scripts to create a demo database (*iqdemo.db*). Many of the queries and code samples in this document use the demo database as a data source.

The demo database contains internal information about a small company (employees, departments, and financial data), as well as product (products), and sales information (sales orders, customers, and contacts).

See the Sybase IQ installation guide for your platform or talk to your system administrator for more information about the demo database.

Accessibility features

This document is available in an HTML version that is specialized for accessibility. You can navigate the HTML with an adaptive technology such as a screen reader, or view it with a screen enlarger.

Sybase IQ 15.1 and the HTML documentation have been tested for compliance with U.S. government Section 508 Accessibility requirements. Documents that comply with Section 508 generally also meet non-U.S. accessibility guidelines, such as the World Wide Web Consortium (W3C) guidelines for Web sites.

Configuring your accessibility tool

You might need to configure your accessibility tool for optimal use. Some screen readers pronounce text based on its case; for example, they pronounce ALL UPPERCASE TEXT as initials, and MixedCase Text as words. You might find it helpful to configure your tool to announce syntax conventions. Consult the documentation for your tool for information on using screen readers.

For information about how Sybase supports accessibility, see Sybase Accessibility at http://www.sybase.com/accessibility. The Sybase Accessibility site includes links to information on Section 508 and W3C standards.

For a Section 508 compliance statement for Sybase IQ, go to Sybase Accessibility at http://www.sybase.com/products/accessibility.

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 Running the Database Server

About this chapter

This chapter describes the available startup switches and the utility programs used to start and stop the Sybase IQ database server. There are other ways to start and stop the database server in addition to these utility programs. For a complete discussion of methods to start and stop the database server, and when each method is appropriate to use, see Chapter 2, “Running Sybase IQ” in the *System Administration Guide: Volume 1*.

Contents

Starting the database server

The database startup utility start_iq starts a Sybase IQ network database server.

The UNIX versions of Sybase IQ provide the script start_iq, and the Windows version of IQ provides start_iq.exe. These scripts verify that your environment is set correctly and start the server with all required switches preset to recommended defaults (along with any switches you add). The start_iq utility also includes some parameters and calculates others. For switches that are specific to your operating system, see the *Installation and Configuration Guide*.

Note The Start Database Server utility in Sybase Central provides an easy graphical interface for starting servers and is the recommended method for starting IQ multiplex servers. See Chapter 2, “Running Sybase IQ” in the *System Administration Guide: Volume 1*.

Syntax

start_iq [*server-options*] [*database-file*
[*database-options*], ...]

Sybase recommends that you specify the -n switch to run Sybase IQ effectively. The -n command-line switch is a server switch that names the server. For example start_iq -n myServer @iqdemo.cfg iqdemo.db. Specifying the -n switch prevents you from unintentionally connecting to the wrong server.

You can list the start_iq switches using the command:

```
start_iq -?
```

Server options parameters

The following table lists the available switches for the start_iq *server options* parameters.

Table 1-1: start_iq server options

Switch	Description
@*filename*	Read in switches from configuration file.
@envvar	Read in switches from environment variable.
-c *cache-size*	Set initial catalog store cache size.
-ca 0	Disable dynamic catalog store cache resizing.
-cc {+|-}	Enable or disable page collection for cache warming.
-ch *size*	Set catalog store cache size upper limit in bytes.
-cl *size*	Set the cache size lower limit in bytes.
-cm *size*	Set Address Windowing Extensions address space size.
-cp	Set search path for classes
-cr {+|-}	Enable or disable cache warming.
-cs	Display cache sizing statistics.
-cv {+|-}	Enable or disable cache warming status messages.
-cw	Enable use of Address Windowing Extensions on Windows 2000, Windows XP, and Windows Server 2003 for setting the size of the database server cache.
-dt *dir*	Directory for temporary files.
-ec *encryption-options*	Enable packet encryption on the network server.
-ep	Displays a dialog box that prompts you for an encryption key to start an encrypted database. Provides extra security by never allowing the encryption key to be seen in clear text.
-es	Allow unencrypted shared memory connections.
-f	Force database to start without transaction log.
-fc *file*	File name of DLL with filesystem full callback function.
-fips	All strong encryption done using FIPS-approved modules. This switch requires the IQ_SECURITY license.
-ga	Automatically shut down database after last connection closed.
-gb *level*	(Windows) Set database process priority class to *level*].
-gc *num*	Set checkpoint timeout period to *num* minutes.
-gd *level*	Set the permission required to start and stop the database.
-ge *size*	(Windows) Sets the stack size for threads that run external functions.
-gf	Disable trigger firing.
-gk *level*	Set the permission required to stop the server.
-gl *level*	Set the permission required to load data.

Switch	Description
-gm *num*	Set the total number of connections that the server can support at one time. If this number is greater than the number that is allowed under licensing constraints, this option has no effect. The value should approximate the number of users expected to connect to the server. Some connections are expected to be idle, while other connections are actively using the database. The default is 10 connections.
-gn *integer*	Set the number of execution threads to be used for the catalog store and connectivity while running with multiple users. Sybase recommends that you set the -gn value to at least 1.5 times the value of -gm.
-gp *size*	Set maximum page size to *size* bytes. *size* must be to the power of 2.
-gr *num*	Set maximum recovery time to *num* minutes.
-gss {*integer* | *integerK* | *integerM*}	Sets the stack size per SQL Anywhere thread in the server. Has no effect on Windows. The default is 64KB. The maximum size is 4MB.
-gt *num*	Set maximum number of physical processors to use (up to licensed maximum).
-gtc *num*	Set maximum processor concurrency (cores and hyperthreads).
-gu *level*	Utility commands permission level: utility_db, all, none, dba.
-iqfrec *dbname*	Open database in forced recovery mode.
-iqgovern *num*	Specify the number of concurrent queries.
-iqmc *size*	Specify the main cache size, in MB. (Overrides default)
-iqsmem *size*	Create a memory pool to increase heap space.
-iqmsgnum *num*	Specifies the number of archives of the old message log maintained by the server. Allowed values are integers 0 – 64 (inclusive). Default value is 0, which means that messages are wrapped in the main message log file. Takes effect only if -iqmsgsz or the IQMsgMaxSize server property is a value other than zero.The IQMsgNumFiles server property corresponds to -iqmsgnum and takes precedence over the value of -iqmsgnum. For information on the IQMsgNumFiles server property, see “sa_server_option system procedure” in Chapter 7, “System Procedures” of *Reference: Building Blocks, Tables, and Procedures*.
-iqmsgsz *size*	Limits the maximum size of the message log. Allowed values are integers 0-2047 (inclusive) in megabytes. The default value is 0, which specifies no limit on the size of the message file.

Switch	Description
-iqmt *num*	Specify the number of threads that IQ can use on a multi-threaded system. The default is `(60*numCPUS)+(2*num_conn)+5`. The minimum value is `2*num_conn+5`.
-iqnotemp *size*	Create a temporary file in place of the defined temporary dbspace. The argument to the switch specifies the file size in MB.
-iqnumbercpus	Override the number of physical CPUs with the number available to IQ. See Chapter 2, "Running Sybase IQ" in the *System Administration Guide: Volume 1*.
-iqpartition	Specify number of partitions in main and temp buffer caches.
-iqsmem *size*	Create a memory pool to increase heap space.
-iqstart *N*	Provide startup diagnostics for dbspaces.
-iqtc *size*	Specify temporary cache size, in MB. (Overrides default.)
-iqtss *size*	Specify the thread stack size, in KB.
-iqwmem *size*	Specify the size, in MB, of a special memory pool that cannot be paged for HP and Sun UNIX platforms.
-k	Instructs Sybase IQ to not collect performance statistics.
-kl *file*	File name for Kerberos GSS-API library.
-kr *realm*	Realm for Kerberos server principal.
-krb	Enable Kerberos authentication. This switch requires the IQ_SECURITY license.
-ks 0	Disable creating performance statistic shared memory.
-ksc *num*	Number of connections to be tracked by the performance statistic. The default is 2.
-ksd *num*	Number or databases to be tracked by the performance statistic. The default is 10.
-m	Truncate transaction log after checkpoint.
-n *name*	Use *name* as the name of the database server. **Note** *There are two -n options.* If *-n* appears after a database file name, the meaning is different. See"Database options parameters" on page 7.
-o *filename*	Output server messages to the specified file.
-oe *filename*	Set file name to log startup errors, fatal errors, and assertions.
-on *size*	Appends ".old" to the log file name and starts a new log file with the original name when the log reaches the specified size.
-os	Specify maximum size of file for server messages.
-ot *file*	File name for copy of message window.

Switch	Description
-p *packet-size*	Set maximum network packet size.
-pc	Compress communication packets.
-pt *size*	Set compression threshold.
-qi	(Windows) Control whether database server tray icon and window appear.
-qp	Do not display messages about performance in the database server window.
-qs	(Windows) Suppress startup error dialogs.
-qw	Do not display database server screen.
-s	Set the syslog facility ID (none, user, daemon, local0,..., local7) [UNIX].
-sb {0 | 1}	Specify how the server reacts to broadcasts on TCP/IP.
-sf *name*	Comma-separated list of features or feature sets to be secured.
-sk *key*	Key allowing overriding of secured features.
-su *password*	Specify password for utility_db, or specify “none” to disable utility_db.
-ti *min*	Client idle time before shutdown:—default 4400 minutes.
-tl *sec*	Default liveness timeout for clients in seconds—default is 120 seconds.
-tmf	Force transaction manager recovery.
-tq *time*	Set quitting time.
-u	Use buffered disk I/O.
-ud	Causes the process to run as a daemon in the root directory [UNIX].
-uf	Specifies the action to take when a fatal error occurs [UNIX].
-ut *min*	Touch temporary files every *min* minutes [UNIX].
-v or -v2	Display database server version.
-x *list*	Comma-separated list of communication links to try.
-xs	Specify server-side Web services communications protocols.
-z	Provide diagnostic information on communication links.
-ze	Display server environment variables.
-zl	Capture most-recently prepared SQL statement for each connection to a database on the server.
-zn *num*	Number of request log files to keep.
-zo	Specify file for logging server requests.
-zoc *file*	Output HTTP Web client procedure debug log to file.
-zr *level*	Enable server request-level logging.

Switch	Description
-zs **{***integer* **|** *integerG* **|** *integerK* **|** *integerM* **}**	Specify maximum size of file for server request logging.

If there is a problem starting the server, start_iq returns a non zero value. If you did not specify a log file after the -o switch on startup, the error is written to the first one of the following that is defined:

- (Windows) *%ALLUSERSPROFILE%\SybaseIQ\IQ15_1\logfiles\*
- (UNIX / Linux) *$IQDIR15/logfiles/*

Database options parameters

The following table lists the available switches for the start_iq *database options* parameters. These options apply only to the preceding database in the command syntax.

Table 1-2: start_iq database options

Switch	Description
-dh	Hide the database from dblocate -d.
-ds *dir*	Specify the directory containing all dbspaces.
-ek *key*	Specify database encryption key.
-m	Truncate transaction log after checkpoint
-n *name*	Name the database **Note** There are two -n switches.The -n switch is positional. If this switch appears after a database file name, the switch is a database switch. Otherwise, -n is a server switch.
-sm *name*	Alternate server name for this database when acting as a read-only mirror.
-sn *name*	Alternate server name for this database.
-r	Read-only mode. Database modifications are not allowed.

Usage

-n server-name Specifies the name of the database server.

database-file Specifies the database file name. If *database-file* is specified without a file extension, Sybase IQ looks for *database-file* with extension *.db*.

If you use a relative path, the path is read relative to the current working directory of the server. You can supply a full path.

On Windows you can supply a path that conforms to the Universal Naming Convention (UNC) format:

```
\\server\volume\path\file.ext
```

Warning! The database file must be on the same machine as the database server. Managing a database file that is located on a network drive can lead to file corruption.

Server command-line switches

@filename Reads in command-line switches from the supplied file.

The file may contain line breaks, and may contain any set of command-line switches. For example, the following Windows command file holds a set of command line switches for a server named iqdemo that allows 10 connections, sets the maximum catalog page size to 4096 bytes, and loads the iqdemo database:

```
# iqdemo.cfg
# ------------------------------------------------
# Default startup parameters for the IQ demo database
# ---------------------------------------------------
-n  iqdemo
-x  tcpip{port=2638}
# The following parameters are also found in the
configuration file
# %IQDIR!%\scripts\default.cfg.  Any parameters not
specified below
# and not in the start up parameter list, will be added
by start_iq
# using default.cfg as a guide.

-c  48m
-gc 20
-gd all
-gl all
-gm 10
-gp 4096
-iqmc 32
-iqtc 24
```

If this configuration file is saved as *c:\config.txt*, the file can be used in a command line as follows:

```
start_iq @c:\config.txt
```

@environment-variable Reads in command-line switches from the supplied environment variable. The environment variable may contain any set of command line switches. For example, the first of the following pair of Windows statements sets an environment variable holding a set of command line switches and loads the sample database. The second statement starts the database server:

```
set envvar= -gp 4096 -gm 15
c:\sybase\IQ-15_1\demo\start_iq -n myserver @envvar
iqdemo.db
```

Note If you have both a file and an environment variable with the value of your @ command-line switch, the result is unpredictable. Use only one of these methods to set a given @ command-line switch.

-c *cache-size* Sets initial memory reserved for caching catalog store pages and other server information. The database server uses extra memory for caching database pages if the memory is set aside in the cache. Any cache size less than 10000 is assumed to be KB (1K = 1024 bytes). Any cache size 10000 or greater is assumed to be in bytes. You can also specify the cache size `nK, nM` or `nP` (1M = 1024 KB), where P is a percentage of the physical system memory.

The default value of -c in the *default.cfg* file and start_iq is 32MB (-c 32M) for Windows platforms, and 48MB (-c 48M) for UNIX platforms. For IQ databases, Sybase recommends that you use this default or set -c to a higher value.

You can use % as an alternative to P, but as most non-UNIX operating systems use % as an environment variable escape character, you must escape the % character. For example, to use 20 percent of the physical system memory, specify:

```
start_iq -c 20%% ...
```

On UNIX operating systems, if the cache size specified with -c is greater than the amount of available memory, the database server uses a maximum cache size that is calculated as follows:

If no `-c` switch is provided (either on the command line or using the start_iq default), the database server computes the initial catalog store cache allocation as follows:

1. The database server uses 32MB as the minimum default cache size.
2. The database server computes a runtime-specific minimum default cache size, which is the lesser of the following items:

- 25% of the physical memory of the machine.
- The sum of the sizes of the main database files specified on the command line. Additional dbspaces apart from the main database files are not included in the calculation. If no files are specified, this value is zero.

3 The database server allocates the greater of the two values computed.

Do not use -c in the same configuration file or command line with -ch or -cl. For related information, see the -ch cache-size option below.

See also the -ca 0 option.

-ca 0 Enforces a static catalog cache size. The zero argument is required.

Ordinarily, the database server automatically takes additional cache as needed. You can disable automatic cache increase due to high server load by using -ca 0 on the command line. The cache size continues to increase if the database server would otherwise run into this error:

```
Fatal Error: dynamic memory exhausted
```

or if the Java VM requires memory that would otherwise lead to a fatal error.

-cc Collects information about database pages to be used for cache warming the next time the database is started. By default, page collection is turned on. When collection is turned on, the database server keeps track of each database page that is requested. Collection stops when the maximum number of pages has been collected, the database is shut down, or the collection rate falls below the minimum value. Note that you cannot configure the maximum number of pages collected or specify the value for the collection rate (the value is based on cache size and database size). Once collection stops, information about the requested pages is recorded in the database so those pages can be used to warm the cache the next time the database is started with the -cr option. Collection of referenced pages is turned on by default.

-ch *cache-size* Limits the catalog store cache that the database server can take during automatic cache growth. By default, the upper limit is approximately the lower of 256MB and 90% of the physical memory of the machine.

You specify the cache-size using the K, M, and P characters as in the -c option. For the meaning and usage of the cache size argument and the K, M, and P characters, see -c cache-size.

In some cases the standard catalog cache size may be too small, for example, to accommodate certain queries that need a lot of parsing. In these cases, you may find it helpful to set -cl and -ch. For example, on 32-bit platforms, try these settings:

```
-cl 128M
-ch 256M
```

Warning! To control catalog store cache size explicitly, you must do *either* of the following, but not both, in your configuration file (*.cfg*) or on the UNIX command line for server startup:

- Set the -c parameter.
- Set specific upper and lower limits for the catalog store cache size using the -ch and -cl parameters.

Specifying different combinations of the parameters above can produce unexpected results.

-cl *cache-size* Sets a lower limit to the catalog store cache during automatic cache growth. By default the lower limit is the initial cache size.

The amount of cache memory available to the catalog store can affect performance, especially when many users are connected. You specify the cache-size using the K, M, and P characters as in the -c option. For the meaning and usage of the cache size argument and the K, M, and P characters, see -c cache-size.

For more information, see Chapter 11, “International Languages and Character Sets” in the *System Administration Guide: Volume 1*.

-cm *size* Specifies the amount of address space allocated for an Address Windowing Extensions (AWE) cache on Windows.The size is the amount of memory, in bytes. Use k, m, or g to specify units of kilobytes, megabytes, or gigabytes, respectively.

-cp Specifies set of directories or jar files in which to search for classes. This option adds directories and jar files to the classpath that the database server builds for launching the Java VM. Use ; to separate directories and jar files.

-cr Reloads (warms) the cache with database pages using information collected the last time the database was run. Cache warming is turned on by default. When a database is started, the server checks the database to see if it contains a collection of pages requested the last time the database was started. If the database contains this information, the previously-referenced pages are then loaded into the cache. Warming the cache with pages that were referenced the last time the database was started can improve performance when the same query or similar queries are executed against a database each time it is started.

-cs Displays cache size information whenever the cache size changes.

-cv Controls the appearance of messages about cache warming. When -cv+ is specified, a message appears in the database server messages window when any of the following cache warming activities occur:

- collection of requested pages starts or stops (controlled by the -cc server option)
- page reloading starts or stops (controlled by the -cr server option)

-cw Enables use of Address Windowing Extensions (AWE) on Windows 2000, Windows XP, and Windows Server 2003 for setting the size of the catalog store cache.

Because Windows 2000, Windows XP, and Windows Server 2003 support Address Windowing Extensions, you can use the -cw option to take advantage of large cache sizes based on the maximum amount of physical memory in the system. Remember, though, that the size of the catalog store cache has much less impact on performance for IQ databases than the IQ main and temporary buffer caches.

Operating system	Maximum non-AWE cache size	Maximum amount of physical memory supported by Windows
Windows 2000 Professional	1.8GB	4GB
Windows 2000 Server	1.8GB*	4GB
Windows 2000 Advanced Server	2.7GB*	8GB
Windows 2000 Datacenter Server	2.7GB*	64GB
Windows XP Home Edition	1.8GB	2GB
Windows XP Professional	1.8GB	4GB

Operating system	Maximum non-AWE cache size	Maximum amount of physical memory supported by Windows
Windows Server 2003, Web Edition	1.8GB	2GB
Windows Server 2003, Standard Edition	1.8GB	4GB
Windows Server 2003, Enterprise Edition	2.7GB*	32GB
Windows Server 2003, Datacenter Edition	2.7GB*	64GB

*You must restart the operating system using the `/3GB` option to use a cache of this size.

When using an AWE cache, almost all of the available physical memory in the system can be allocated for the cache.

If you can set a catalog store cache of the desired size using a non-AWE cache, Sybase recommends that you do so, because AWE caches allocate memory that can only be used for the catalog store. This means that while the database server is running, the operating system and the IQ store caches cannot use the memory that is allocated for the catalog store cache.

AWE caches do not support dynamic cache sizing. Therefore, if an AWE cache is used and you specify the -ch or -cl options to set the upper and lower cache size, they are ignored.

To start a database server with an AWE cache:

- Have at least 130MB of memory available on your system.
- If your system has between 2GB and 16GB of memory, add the `/3GB` option to the Windows boot line in the “[operating systems]” section of the *boot.ini* file.

  If your system has more than 16GB of memory, do not add the `/3GB` option to the Windows boot line in the “[operating systems]” section of the *boot.ini* file because Windows cannot address memory beyond 16GB.
- If your system has more than 4GB of memory, add the `/PAE` option to the Windows boot line in the “[operating systems]” section of the *boot.ini* file.
- Grant the “Lock pages in memory” privilege to the user ID under which the server is run. The following steps explain how to do this on Windows 2000.

a Log in to Windows as Administrator.

b Select Start > Settings > Control Panel.

c Open the Administrative Tools folder.

d Double-click Local Security Policy.

e In the left pane, open Local Policies.

f In the left pane, double-click User Rights Assignment.

g In the right pane, double-click the Lock Pages In Memory policy.

h In the Local Security Policy Setting dialog, click Add.

i Select the user ID and click Add.

j In the Local Security Policy Setting dialog, click OK.

k Restart the computer for the setting to take effect.

If you specify the -cw option and the -c option on the command line, the database server attempts the initial cache allocation as follows:

1. The AWE cache is no larger than the cache size specified by the -c option. If the value specified by the -c option is less than 2MB, AWE is not used.
2. The AWE cache is no larger than all available physical memory less 128MB.
3. The AWE cache is no smaller than 2MB. If this minimum amount of physical memory is not available, an AWE cache is not used.

When you specify the -cw option and do not specify the -c option, the database server attempts the initial cache allocation as follows:

1. The AWE cache uses 100% of all available memory except for 128MB that is left free for the operating system.
2. The AWE cache is no larger than the sum of the sizes of the main database files specified on the command line. Additional dbspaces apart from the main database files are not included in the calculation. If no files are specified, this value is zero.
3. The AWE cache is no smaller than 2MB. If this minimum amount of physical memory is not available, an AWE cache is not used.

When the server uses an AWE cache, the catalog cache page size is at least 4KB, and dynamic cache sizing is disabled. On 64-bit Windows platforms, the cache page size is at least 8KB.

For more information about dynamic cache sizing, see the -ch and -cl server options.

-dt *dir* Specifies the directory where temporary files are stored.

-ep Displays a dialog box that prompts user to enter encryption key for running a strongly encrypted database. This option provides an extra measure of security by never allowing the encryption key to be seen in clear text. For a strongly encrypted database, you must specify either -ep or -ek, but not both. The command fails if you do not specify a key for a strongly encrypted database.

The engine cannot be a Windows service, or it must be a Windows service with the interact with desktop option turned on.

When used with supported tools, this option always prompts the user for the encryption key, even if a key is not necessary. If you know that a key is not necessary, click Cancel to continue when the dialog box prompt appears.

-es Allows unencrypted connections over shared memory.

-f Forces the database server to start after the transaction log has been lost. This option is for use in recovery situations only. If there is no transaction log, the database server performs a checkpoint recovery of the database and then shuts down—it doesn't continue to run. You can then restart the database server without the -f option for normal operation.

If there is a transaction log in the same directory as the database, the database server performs a checkpoint recovery, and a recovery using the transaction log, and then shuts down—it doesn't continue to run. You can then restart the database server without the -f option for normal operation.

-fc *file* Specifies the file name of a DLL (or shared object on UNIX) containing the File System Full callback function.

-fips Specifying this option forces all strong database encryption to use FIPS-approved algorithms.

-ga Specifying this switch causes the database server to automatically shut down after the last database closes.

-gb *level* Sets the database process priority class to *level*. Level must be one of idle, normal (the default), high, or maximum. The commonly used settings are normal and high. The value idle is provided for completeness, and maximum may interfere with the running of your computer.

-gc *num* Sets the maximum number of minutes the database server runs without doing a checkpoint on each database. The default value is 20 minutes.

-gd *level* Sets the permission level required to start a database on the server, or to stop a database:

- **DBA** Only users with DBA authority can start or stop a database.
- **ALL** All users can start a database or stop a database (default in start_iq and *default.cfg*). Use this setting so that the DBA does not need to issue START DATABASE commands. Users must still have permission to access a database once they have started it.)
- **NONE** Starting a database from Interactive SQL is not allowed. You can still stop the server and database by running stop_iq on UNIX or Shutdown on Windows.

You can use either uppercase and lowercase syntax for the options.

-ge *size* (Windows) Sets the stack size for threads running external functions, in bytes. The default is 16384 (16KB).

-gf Disables triggers.

-gk *level* Sets the permission level required to stop the database server:

- **DBA** Only users with DBA authority can use dbstop to stop the server (the default).
- **ALL** All users can use dbstop to stop the server.
- **NONE** The server cannot be stopped using dbstop.

You can use either uppercase and lowercase syntax for the options.

-gl *level* Sets the permission required to load data using LOAD TABLE.

The LOAD TABLE statement reads files from the database server machine.

To control access to the file system using these statements, the -gl command-line switch allows you to control the level of database permission that is required to use these statements.

- **DBA** Only users with DBA authority can load data.
- **ALL** All users can load data.
- **NONE** Data cannot be loaded.

Both uppercase and lowercase syntax are acceptable.

The default settings are all for servers started with start_iq and dba for other servers. Sybase recommends that, for consistency with earlier versions, you use the all value on all systems. The all setting is used in the *iqdemo.cfg* and *default.cfg* configuration files. See Table 2-1 in Chapter 2, “Running Sybase IQ” in the *System Administration Guide: Volume 1*.

-gm *num* Sets the total number of connections that the server can simultaneously support. If this number is greater than the number that is allowed under licensing and memory constraints, this switch has no effect. The value should approximate the number of users expected to connect to the server. Some connections are expected to be idle, while other connections are actively using the database. The default varies by machine capacity, but 15 is recommended.

The database server allows one extra DBA connection above the connection limit to allow a DBA to connect to the server and drop other connections in an emergency.

-gn *num* Sets the number of execution threads that will be used for the catalog store and connectivity while running with multiple users. This parameter applies to all operating systems and servers.

Each connection uses a thread for each request, and when the request is completed, the thread is returned to the pool for use by other connections. As no connection can have more than one request in progress at one time, no connection uses more than one thread at a time.

An exception to this rule is if a Java application uses threads. Each thread in the Java application is a database server execution thread.

On Windows, specify this parameter in the start_iq command. To calculate its value, use:

```
gn_value >= gm_value * 1.5
```

Sybase recommends that you set the -gn value to at least 1.5 times the value of -gm. Specify a minimum of 25. The total number of threads must not exceed a platform-specific maximum; see “-iqmt num” on page 19 for details.

The start_iq utility sets this parameter. See the *Installation and Configuration Guide* for your platform for more information.

-gp *size* Sets the maximum page size allowed, in bytes, for the catalog store. The size specified must be 1024, 2048, 4096, 8192, 16384, or 32768. The minimum page size on all UNIX platforms is 2048 bytes. Sybase highly recommends that you set the -gp switch to at least 4096.

-gr *num* Sets the maximum length of time, in minutes, that the database server takes to recover from system failure.

See “RECOVERY_TIME option” in Chapter 2, “Database Options” in *Reference: Statements and Options*.

-gss { integer | integerK | integerM } Sets the stack size for each SQL Anywhere thread in the server. Has no effect on Windows operating systems. The default stack size per SQL Anywhere thread is 64KB. The maximum size is 4MB.

-gt *num* Sets the maximum number of physical processors that can be used (up to the licensed maximum). This option is only useful on multiprocessor systems.

-gtc *num* Controls the maximum processor concurrency (cores and hyperthreads) that the database server allows.

-gu *level* Sets permission levels for commands such as CREATE DATABASE and DROP DATABASE. Set the level to one of: utility_db, all, none, dba.

The utility_db level restricts the use of these commands to only those users who can connect to the utility database. The all, none, and dba levels respectively permit all users, no users, or connected users with DBA authority to execute utility commands.

To connect to the utility database, specify utility_db as the database name (that is, there is no database file) and specify the utility database password (stored in the file *util_db.ini*).

-iqgovern *num* Sets the number of concurrent queries allowed by the server. The number of concurrent queries is not the same as the number of connections. -iqgovern can help Sybase IQ optimize paging of buffer data out to disk and avoid overcommitting memory. The default value of this switch is equal to 2 times the number of CPUs on your machine, plus 10. You may find that another value, such as 2 times the number of CPUs plus 4, provides better throughput, especially when large numbers of users are connected.

-iqmc *size* Specifies the main IQ store cache size in MB. The switch overrides default of 16MB, or the value, if any, set by the MAIN_CACHE_MEMORY_MB database switch. Applies to all databases started from the time the IQ server is started until the IQ server is shut down. In other words, if you start one database at server startup and another later, you need 2 * -iqmc available for the main cache. In general, Sybase recommends that you do not run multiple databases with a Sybase IQ server.

-iqmsgnum *num* Specifies the number of archives of the old message log maintained by the server. The value of -iqmsgnum takes effect only if the IQMsgMaxSize server property or the -iqmsgsz server startup switch is non zero. The IQMsgNumFiles server property corresponds to -iqmsgnum and takes precedence over the value of -iqmsgnum.

-iqmsgnum is an integer 0 – 64 (inclusive). The default value is 0, which means that messages are wrapped in the main message log file.

A -iqmsgnum value *n* greater than 0 means that the server maintains *n* message log archives. For example, this command specifies that the server maintain 3 archives of the message log:

```
start_iq -n iqdemo iqdemo.db ... <other options> ...
-iqmsgsz 100 -iqmsgnum 3
```

For information on the IQMsgNumFiles server property, see "sa_server_option system procedure" in Chapter 7, "System Procedures" in *Reference: Building Blocks, Tables, and Procedures*.

For information on message log management, see "Message logging" in Chapter 1, "Overview of Sybase IQ System Administration" in the *System Administration Guide: Volume 1*.

-iqmsgsz *size* Limits the maximum size of the message log. -iqmsgsz is an integer 0-2047 (inclusive), in MB. The default value is 0, which specifies that there is no limit on message log size. The IQMsgMaxSize server property corresponds to the -iqmsgsz server switch and takes precedence over the value of -iqmsgsz.

A -iqmsgsz value *n* greater than 0 means that the message log can grow up to *n* megabytes in size. For example, the following command limits the size of the message log to 100MB:

```
start_iq -n iqdemo iqdemo.db ... <other options> ...
-iqmsgsz 100
```

For information on the IQMsgMazSize server property, see "sa_server_option system procedure" in Chapter 7, "System Procedures" in *Reference: Building Blocks, Tables, and Procedures*.

For information on message log management, see "Message logging" in Chapter 1, "Overview of Sybase IQ System Administration" in the *System Administration Guide: Volume 1*.

-iqmt *num* Specifies the number of Sybase IQ threads to create. The default is 60 per CPU for the first four CPUs and 50 per CPU for the remainder, plus connection threads. For example, on a system with 12 CPUs and 10 connections, `60*4 + 50*(numCPUs - 4) + 2*(numConnections + 2) + 1 = 665`. The minimum value is `2*num_conn+1`. The total number of threads (-iqmt plus -gn) cannot exceed 4096 on 64-bit platforms, or 2048 on 32-bit platforms. The default -iqtss setting should be adequate to support these maximum numbers of threads.

-iqnotemp *size* Creates a temporary file in place of the defined temporary dbspace. *size* is file size in MB.

-iqnumbercpus *num* Specifies the number of CPUs available to IQ, overriding the physical number of CPUs for resource planning purposes. The value of -iqnumbercpus defaults to the total number of CPUs, but the range of available values is 1 – 128.

Sybase recommends that you use -iqnumbercpus only on:

- Machines with Intel CPUs and hyperthreading enabled, setting -iqnumbercpus to the number of CPUs available
- Machines where an operating system utility has been used to restrict Sybase IQ to a subset of the CPUs within the machine

Setting -iqnumbercpus higher than the number of available CPUs may affect performance.

-iqpartition Specifies the number of partitions in the IQ main and temp buffer caches. Must be a power of 2. Allowed values are: 0 (default), 1, 2, 4, 8, 16, 32, 64. By default, IQ computes the number of partitions automatically as *number_of_cpus/8*, rounded to the nearest power of 2, up to a maximum of 64. You may be able to improve performance by adjusting the number of cache partitions. The -iqpartition switch sets this value for an IQ server, and overrides the value set by the Cache_Partitions database option. See "CACHE_PARTITIONS option" in Chapter 2, "Database Options" in the *Reference: Statements and Options*, and "Managing lock contention" in Chapter 10, "Transactions and Versioning" in the *System Administration Guide: Volume 1* for more information on whether you should set one of these options, and factors that affect them.

-iqsmem *size* Creates a memory pool (in MB) to increase heap space.

-iqstart *N* Provides startup diagnostics for dbspaces. The input parameter *N* is a number value that represents an integer bit mask. You may combine values to provide more than one feature. Output generated before the IQ message file is generated goes to the console. The -z startup switch provides additional startup and connection information.

The allowed values are as follows:

- **N=1** Returns basic information about the file names from SYSIQFILES that are used when opening the dbspace. It then displays the fully qualified names used. You can use this option to create a record of the files in use by the database in the IQ message file.

- **N=2** Stops after the transaction log replay before executing RecoveryComplete allowing you to examine the database without opening it all the way. You can combine N=2 with other options. In certain modes using N=2 may rewrite the commit_identity, but does not otherwise modify the database in a permanent manner—the checkpoint that would commit the recovery actions is not allowed to complete. All recovery actions reexecute the next time the database is opened.
- **N=4** Returns full diagnostic information, including all rows of SYSIQFILE, the subset of file names selected if the database is a multiplex database, the fully resolved file names, each individual dbspace file header block, the database_identity, the commit_identity, each checkpoint log entry, and each transaction log entry.
- **N=8** Allows the file paths in SYSIQFILE to be overridden. Instead of the SYSIQFILE values, the file names *iqmsg.iqmsg, iqmain_1, iqmain_2, ..., iqtemp_1, iqtemp_2,* and so on. will be used. These may be links and must be in the same directory as the *.db* file. You may use a link to the actual *.db* file but if the server is given a link to a *.db* file that uses a transaction log relative to the database, the server looks for the transaction log relative to the link, rather than to the database. In this case, create a link for the transaction log also.

-iqtc *size* Specifies IQ temporary store cache size in MB. Overrides default of 8MB, or value, if any, set by TEMP_CACHE_MEMORY_MB database option. Applies to all databases started from the time the IQ server is started until the IQ server is shut down. In other words, if you start one database at server startup and another later, you need 2 * -iqtc available for the temp cache. In general, Sybase recommends that you do not run multiple databases with a Sybase IQ server.

-iqtss *size* Sets the stack size of the internal execution threads in the server. The default value is 200KB for 32-bit platforms, and 512KB for 64-bit platforms, which is the recommended value for each platform. However, some complex queries may return an error indicating that the depth of the stack exceeded this limit, so the DBA may need increase this value. On 32-bit platforms, the best setting depends on the available memory, number of CPUs, number of connections and type of queries run. On 64-bit platforms, the value 1000 is reasonable because the memory of 64-bit systems is not constrained as it is in 32-bit systems. Do not choose a value less than the default or you may find that some queries cannot run.

-iqwmem *size* Creates a pool of “wired” memory on HP and Sun UNIX systems. This memory is locked down so it cannot be paged by the operating system. Specify the memory size, in MB. Use this switch only if you have enough memory to dedicate for this purpose. Otherwise, you may cause serious performance degradation.

-k Instructs the database server to not collect Performance Monitor statistics.

-kl *file* Specifies the file name of the Kerberos GSS-API library (or shared object on UNIX) and enables Kerberos authenticated connections to the database server.

-kr *realm* Specifies the realm of the Kerberos server principal and enable Kerberos authenticated connections to the database server. Normally, the principal used by the database server for Kerberos authentication is server-name@default-realm, where default-realm is the default realm configured for the Kerberos client. Use this option if you want the server principal to use a different realm than the default realm, in which case the server principal used is server-name@server-realm.

-krb Enables Kerberos-authenticated connections to the database server.

-ksc *num* Specifies the maximum number of connections that the Performance Monitor can monitor. The default number of connections is two.

-ksd *num* Specifies the maximum number of databases that the Performance Monitor can monitor. The default number of databases is 10.

-ks *0* Disables the creation of shared memory that the Performance Monitor uses to collect counter values from the database server.

-m Deletes the transaction log when a checkpoint is done, either at shutdown or as a result of a checkpoint scheduled by the server. This switch lets you automatically limit the growth of the transaction log. Checkpoint frequency is still controlled by the CHECKPOINT_TIME and RECOVERY_TIME options (which also can be set in the command line).

The -m server switch is useful if you are processing high-volume transactions requiring fast response times, and the contents of the transaction log are not being relied upon for recovery or replication.

Warning! When you select the -m server switch, there is no protection against media failure on the device that contains the database files. Additionally, do not use the -m switch with databases that are being replicated, as replication inherently relies on transaction log information. For this reason, never use the -m switch on a multiplex database.

To avoid database file fragmentation, when you use this switch place the transaction log on a separate device or partition from the database itself.

If you start the server with the -m switch, you cannot create a database.

See "The transaction log file" in Chapter 4, "Managing System Resources" in the *Performance and Tuning Guide*.

-n *name* Sets the name of the database server. By default, the database server receives the name of the database file with the path and extension removed. For example, if you start a server on the file *c:\sybase\IQ-15_1\demo\iqdemo.db* and do not specify the -n switch, the name of the server is iqdemo.To avoid using the default name, always specify a server name.

Each server name must be unique across the local area network (domain). This prevents you from unintentionally connecting to the wrong server. The host name and port number combination does not uniquely identify the server.

The server name is interpreted according to the character set of the machine, as no database collation exists at startup time. The server name must be a valid identifier. Long server names are truncated to 40 characters. On NetBIOS, 16 characters is the maximum length. For multiplex server names, 30 characters is the maximum.

The server name specifies the name to be used on client application connection strings or profiles.

Note There are two -n switches. If -n appears after a database file name, the switch is a database option. Otherwise, the switch is a server option. For example, in the following start_iq command line, the first -n indicates a server name and the second -n, which follows the database file name *mydb.db*, indicates a database name:

```
start_iq -n svrname mydb.db -n dbname
```

The server name must be used on the connect statement to specify the server you are connecting to. In all environments, there is always a default database server that is used if no server name is specified, provided at least one database server is running on the system.

Multiple database servers with the same name are not allowed to run on TCP/IP anywhere on the network, even on separate ports.

-o *filename* Prints all server message window output to a file, in addition to displaying the output on the screen. You can use this option to specify a nondefault name for the server log file, and keep using the same file after restarting the server. Be careful that you do not fill this file.

Note If the -o file is located within a file system that fills up, then the IQ server stops responding. Once this condition exists, the only way to bring down the server is to kill it.

-oe Specifies a file name to log startup errors, fatal errors, and assertions. Each line in the output log file is prefixed with the date and time.

-on *size* Specifies a maximum size for the database server message log, after which the file is renamed with the extension .old and a new file is started.

-os *size* Limits the size of the log file used by the -o switch. The default is no limit. You can specify G, K, and M units using either in lowercase or uppercase. If you do not specify units, any integer less than 10,000 is assumed to be in kilobytes, and any integer 10,000 or greater is assumed to be in bytes.

-ot *file* Truncates the database server message log file and appends output messages to it.

-p *packet-size* Sets the maximum size of communication packets. The range of values is 300 – 16,000 bytes and the default is 1460 bytes. See “CommBufferSize connection parameter [CBSize]” in Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

-pc Compresses all connections except for same-computer connections.

-pt *size* Increases or decreases the size limit at which packets are compressed. -pt takes an integer value representing the minimum byte-size of packets to be compressed. The default is 120 bytes, which should be appropriate in most situations. Sybase recommends that you do not use a value less than 80.

Under some circumstances, changing the compression threshold may improve performance of a compressed connection by allowing you to compress packets only when compression will increase the speed at which the packets are transferred.

If the client and server specify different compression threshold settings, the client setting takes precedence.

-qi (Windows servers) Controls whether the database server tray icon and window appear. If you use this option, there is no visual indication that the server is running, other than possible startup error dialogs. You can use either (or both) the -o or -oe logs to diagnose errors.

-qp Disables display of messages about performance in the database server window. Suppressed messages include:

- No unique index or primary key for table '*table_name*'
- Database file "*mydatabase.db*" consists of *nnn* fragments

-qs (Windows servers) Suppresses startup error dialogs. Startup errors include:

- Could not open/read database file: <*database file*>
- A database server with that name has already started

On Windows platforms, if the server is not being autostarted, these errors appear in a dialog and must be cleared before the server stops.

If there is an error loading the language DLL, no dialog appears if -qs was specified on the command line, and not in *@environment-variable* or *@filename* syntaxes. This error is logged in the Windows Application Event Log.

Usage errors are suppressed if -qs is on the command line, but not in *@filename* or *@environment-variable* expansion.

-qw Suppresses the database server window (Windows platforms) and displays messages on the console (non-Windows platforms).

-s <*id*> (UNIX servers) Sets the system user ID used in messages to the syslog facility. The default is user, which uses the user ID for the database server process. A value of none prevents any syslog messages from being logged.

-sb { 0 | 1 } Specifies how the server reacts to broadcasts on TCP/IP.

-sb 0 means the server does not start up any TCP/UDP broadcast listeners. In addition to forcing clients to use the DoBroadcast=NONE and HOST= options to connect to the server, -sb 0 causes the server to be unlisted when using dblocate.

-sb 1 means the server does not respond to broadcasts from dblocate, but leaves connection logic unaffected. You can connect to the server by specifying LINKS=tcpip and ENG=<*name*>.

-sf *feature* This option allows you to enable and disable features for a database server. The *feature-list* is a comma-separated list of feature names or feature sets to secure for the database server.

-sk *key* Specifies a key that can be used to enable features that are disabled for the database server.

-su *password* This option specifies the initial password for the DBA user in the utility database (utility_db). The password is case-sensitive. You can specify none for the password to disable all connections to the utility database.

-ti *minutes* Disconnects connections that have not submitted a request for the number of minutes you specify. The default is 4400 (72 hours), so that a user running a long query will not be logged off over a long weekend. A client machine in the middle of a database transaction holds locks until the transaction is ended or the connection is terminated. By disconnecting inactive connections, -ti frees these locks. The -ti switch does not disconnect clients that use the shared memory communications link. Using -ti has no effect on connections to a local server using shared memory. Setting the value to zero disables checking of inactive connections, so that no connections are disconnected.

You can use the IDLE connection parameter to set timeout values for individual connections. For more information, see “Idle connection parameter [IDLE]” in Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

-tl *seconds* Sets the liveness timeout period for the server. A liveness packet is sent periodically across a client/server TCP/IP communications protocol to confirm that a connection is intact. If the server runs for a liveness timeout period (default 2 minutes) without detecting a liveness packet, the communication is severed and the server drops any connections associated with that client. UNIX non threaded clients and TDS connections do not perform liveliness checking.

The -tl switch on the server sets the liveness timeout for all clients that do not specify a -tl switch.

When there are more than 200 connections, the server automatically calculates a higher LivenessTimeout value based on the stated LivenessTimeout value. This enables the server to handle a large number of connections more efficiently.

Clients send liveness packets at an interval of between one third and two thirds of the LivenessTimeout value on each idle connection. Large numbers of liveness packets are not sent at the same time. If liveness packets take a long time to send (depending on the network, the machine's hardware, and the CPU and network load on the machine), liveness packets may be sent after two thirds of the LivenessTimeout. A warning appears in the server console if the liveness sends take a long time. If this warning occurs, consider increasing the LivenessTimeout value.

-tmf Forced transaction manager recovery. Used during recovery of distributed transactions when the distributed transaction coordinator is not available.

-tq *time* Shuts down the server at a specified time. The format for the time is HH:MM (24-hour clock), and can be preceded by an optional date. If you do specify a date, enclose the date and time in double quotes and use the format “YYYY/MM/DD HH:MM”.

-u Opens files using the operating system disk cache, rather than opening files using the database cache only.

-ud (UNIX servers) Causes the process to run as a daemon in the root directory. Sybase recommends that you do not use this switch in IQ servers.

-uf (UNIX servers) Specifies the action to take when a fatal error occurs.

- **abort** – the UNIX abort function is called, and a core file is generated.
- **default** – the database server behaves in the same manner as abort in all cases, except when a device-full fatal error occurs. In this case, it behaves in the same manner as defunct. This action prevents the system from trying to write a core file on a full device. This is the default behavior.
- **defunct** – the database server continues running and does not call abort. Any new connection attempts made to the database server receive the SQL error of the original fatal error.

-ut min (UNIX servers) Causes the server to touch catalog store temporary files at intervals specified by *min*.

-v Displays the database server version in a message box (Windows) or in a version string (UNIX / Linux).

-x *list* Specifies server-side network communications protocols.

list is a comma-separated list of tcpip or namedpipes settings. For example:.

For example,

```
-x tcpip,ipx
```

allows only TCP/IP and IPX communications.

The default is to try all settings supported by the database server on your operating system.

For some protocols, you can provide additional parameters, in this format:

```
-x tcpip(PARM1=value1;PARM2=value2;...)
```

For UNIX, quotation marks are required if more than one parameter is supplied:

```
-x "tcpip(PARM1=value1;PARM2=value2;...)"
```

For a description of available parameters, see Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

-xs Specifies server-side Web services communications protocols.

-xs { **all** | **none** | *web-protocols* } …

web-protocols: { [**http** | **https** | **http_fips**] *parmlist* },… *parmlist*: **(** *parm*=*value*;…**)**

Use the -xs option to specify the Web protocols you want to use to listen for client connection broadcasts. If you specify one or more protocols, the server attempts to listen for client requests using the specified protocol(s).

If you do not specify the -xs option, the server does not attempt to listen for Web requests.

Regardless of the settings you choose for the -xs option, the server always listens for connection broadcasts using the shared memory protocol. You can specify any of the following:

- **ALL** Listen for connection attempts by the client using all communications protocols that are supported by the server on this platform, including shared memory.
- **HTTP** Listen for connection attempts by the client using the HTTP protocol. The default port on which to list is 80.
- **HTTPS** Listen for connection attempts by the client using the HTTPS protocol. The default port on which to list is 443.
- **NONE** Listen for connection attempts by the client using only the shared memory protocol. This is the default.

For a description of available parameters, see “Network communications parameters” in Chapter 4, “Connection and Communication Parameters”in the *System Administration Guide: Volume 1*.

For UNIX, you must use quotation marks if you are supplying more than one parameter:

```
-xs "http(PARM1=value1;PARM2=value2;...)"
```

This command allows only shared memory and TCP/IP communications:

```
start_iq web.db -xs http(port=80)
```

See “CommLinks connection parameter [Links]” in Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

-z Provides diagnostic information about communication links on startup. Use this switch only when you are troubleshooting problems.

-ze Displays database server environment variables in the database server messages window.

-zl Enables capturing of the most recently-prepared SQL statement for each connection to a database on the server.

Alternatively, you can enable this feature using the remember_last_statement option of the sa_server_option system procedure. You can obtain the most recently-prepared SQL statement for a connection using the LastStatement property function. The sa_conn_activity stored procedure allows you to obtain the most recently-prepared SQL statement for all current connections to the database on the server.

For stored procedure calls, only the outermost procedure call appears.

-zn *num* Specifies the number of request log file copies to retain. Used in conjunction with -zs.

-zo *file* Redirects request-level logging information to a file separate from the regular log file. Request-level logging is turned on using the -zr switch. The -zo switch directs the output from this file to a separate file from that specified on a -o switch. This switch also prevents request-level logging from being displayed in the console.

-zoc *file* Redirects HTTP Web service client procedure debug log to a file.

-zr *level* Enables request-level logging of operations:

- **All** Logs all SQL statements and other requests to the server.
- **None** Turns off SQL statement logging. This is the default.
- **SQL** Logs the following types of requests only:
    - CONTROL_START_DATABASE

- CONTROL_STOP_ENGINE
- CONTROL_STOP_DATABASE
- STMT_PREPARE
- STMT_EXECUTE
- STMT_EXECUTE_IMM
- STMT_EXECUTE_ANY_IMM
- SQL_OPTION_SET
- BACKUP
- DELETE_FILE
- COMMIT
- ROLLBACK
- PREPARE_TO_COMMIT
- CONNECT
- DISCONNECT
- BEGIN_TRANSACTION
- STMT_DROP
- CURSOR_OPEN
- CURSOR_EXPLAIN
- CURSOR_CLOSE
- CURSOR_RESUME
- Errors

-zr also prevents request-level logging from appearing in the console. See "Logging server requests" in Chapter 14, "Troubleshooting Hints" of *System Administration Guide: Volume 1*. To correlate connection information in the -zr log file with that in the *.iqmsg* file, see "Correlating connection information," and for information about reading the -zr log output, see "Request log file analysis," both in Chapter 14, "Troubleshooting Hints" in the *System Administration Guide: Volume 1*.

See also “Request logging” in the SQL Anywhere documentation at S*QL Anywhere 11.0.1 > SQL Anywhere Server - SQL Usage > Monitoring and Improving Database Performance > Improving database performance > Other diagnostic tools and techniques*.

See also “-zo file” and “-zs { integer | integerG | integerK | integerM } …” command-line switches.

-zs { ***integer*** **|** ***integerG*** **|** ***integerK*** **|** ***integerM*** **} …** Limits the size of the request-level logging file. Request-level logging is turned on using the -zr switch, and redirected to a separate file using the -zo switch. You can limit the size of the file using the -zs switch.

You can specify G, K, and M units using either uppercase or lowercase. If you do not specify units, any integer less than 10 000 is assumed to be in kilobytes, and any integer 10,000 or greater is assumed to be in bytes.

When the request log file reaches the size specified by either the -zs option or the sa_server_option system procedure, the file is renamed with the extension *.old* appended (replacing an existing file with the same name if one exists). The request-level log file is then restarted.

By default there is no limit. The value is in kilobytes.

The following example shows how the -zs option is used to control log file size. Suppose you start a database server with the following options on the command line:

```
-zr all -zs 10 -zo mydatabase.log
```

A new log file *mydatabase.log* is created. When this file reaches 10K in size, any existing *mydatabase.old* files are deleted, *mydatabase.log* is renamed to *mydatabase.old*, and a new *mydatabase.log* file is started. This process is repeated each time the *mydatabase.log* file reaches 10K.

Database options

-dh Makes a database undetectable when the Server Enumeration utility (dblocate) is run against the server.

-ds Specifies the directory where the dbspaces for the database are located. When a dbspace directory is specified, the database server only searches this directory for dbspaces.

–ec Uses Transport Layer Security or simple encryption to encrypt all native Sybase IQ packets (DBLib, ODBC, and OLE DB) transmitted to and from all clients. TDS™ (Tabular Data Stream) packets are not encrypted.

-ec *encryption-options* …

encryption-options:

```
 { NONE |
      SIMPLE |
TLS ( TLS_TYPE=cipher;
[FIPS={Y|N};]
IDENTITY=server-identity-filename;
IDENTITY_PASSWORD=password ) } , …
```

See "-ec server option" in the SQL Anywhere documentation at *SQL Anywhere 11.0.1 > SQL Anywhere Server - Database Administration > Starting and Connecting to Your Database > The database server > Database server options*.

see also "Encryption connection parameter [ENC]" in Chapter 4, "Connection and Communication Parameters" in the *System Administration Guide: Volume 1*.

-ek Provided after the file name of a strongly encrypted database. Requires the key value as an argument to start an encrypted database. The key value is a string, including mixed cases, numbers, letters, and special characters. If you have a strongly encrypted database, you must provide the encryption key to use the database or transaction log; if you do not, the command fails. For a strongly encrypted database, you must specify either -ek or -ep, but not both.

-m Truncates (deletes) the transaction log when a checkpoint is done, either at shutdown or as a result of a checkpoint scheduled by the server. This provides a way to automatically limit the growth of the transaction log. Checkpoint frequency is still controlled by the CHECKPOINT_TIME and RECOVERY_TIME options (also definable on the command line).

The -m option is useful where high volume transactions requiring fast response times are being processed, and the contents of the transaction log are not being relied upon for recovery or replication. When this option is selected, there is no protection provided against media failure on the device containing the database files.

To avoid database file fragmentation, Sybase recommends that if you use the -m option, you place the transaction log on a separate device or partition than the database itself.

This option is the same as the -m server option, but applies only to the current database or the database identified by the *database-file* command-line variable.

Note Do not use the -m option with databases that are being replicated, as replication inherently relies on transaction log information. For this reason, *never* use the -m option on a multiplex database.

-n name Provides an alternate name, or nickname, for the database. Using a nickname simplifies connections. For Open Client, the -n nickname must be the same as the entry in the *interfaces* file.

Since a database server can load several databases, the database name is used to distinguish the different databases. However, Sybase strongly recommends that you run only one database on an IQ server. If you must run two databases, start two IQ database servers on different ports.

By default, the database receives as a name the file name with the path and extension removed. For example, you start a server on *c:\sybase\IQ-15_1\demo\iqdemo.db* and do not specify the -n option, then the name of the database is iqdemo. To avoid using the default name, always specify a server name.

For naming conventions, see the -n server option.

Note There are two -n switches. If -n does not follow a database file name, the option names the server. If -n appears after a database file name, the switch is a database switch.

-sm Provides an alternate database server name that can be used to access the read-only mirror database. The alternate-server-name is only active when the database server is acting as mirror for the database. By using the -sm and -sn command-line options, an application can always connect to the database on the primary or the mirror server, without knowing which physical server is acting as primary or mirror.

-sn Provides an alternate server name for a single database running on a database server. The database server can be configured to listen for more than one server name for a particular database server. Server names other than the real server name are called alternate server names, and are specific to a particular database running on the database server. Clients using the alternate server name to connect can only connect to the database that specified the alternate server name. Alternate server names must be unique on the network; otherwise, the database fails to start. If the database is started in the server command and the alternate server name is not unique, the server fails to start.

-r Forces all databases that start on the database server to be read-only. No changes to the database are allowed: the database server doesn't modify the database file.

Recovery options

-iqfrec dbname Marks the specified database as in use and restores the IQ portion of the database to its last known consistent state. Do not use -iqfrec during normal operations; use it only while force-recovering a database. The *dbname* must be the physical database name, not a logical name or nickname.

Note The option -iqfrec applies only to the IQ part of the database, not to the catalog store. -iqfrec does not enable a forced recovery on the SQL Anywhere part of the database (the catalog store).

Follow correct procedures when using -iqfrec. See Chapter 13, “System Recovery and Database Repair” in the *System Administration Guide: Volume 1*.

See also

Chapter 2, “Running Sybase IQ” in the *System Administration Guide: Volume 1*

Stopping the database server

There are several ways to stop the database server, some of which are platform-specific. This section describes the two operating system level server shutdown utilities dbstop and stop_iq. For a full discussion of all of the server shutdown methods and when you need to stop the database server, see the section “Stopping the database server” in Chapter 2, “Running Sybase IQ” in the *System Administration Guide: Volume 1*.

dbstop

dbstop stops a database server. The Stop utility is a command-line utility only, available on both UNIX and Windows platforms.

On Windows platforms, you can quickly stop a database server by clicking Shutdown on the server window or by right-clicking the server icon in the taskbar and choosing Exit.

Syntax

dbstop [*options*] *server-name*

Parameters

Table 1-3: dbstop options

Option	Description
@filename	Read in options from the specified environment variable or configuration file.
server-name	Server name of a running server.
-c *“keyword=value; ...”*	Connection parameters.
-d	Do not stop the database server. Instead, only stop the database specified in the connection string.
-o *filename*	Log output messages to a file.
-q	Quiet mode—do not print messages.
-x	Do not stop if there are active connections.
-y	Stop without prompting even if there are active connections.

Usage

In UNIX, dbstop can shut down a server on any node on the network. You must specify a *server-name*, as well as any connection parameters you specified when you started the server. Without the proper connection parameters, dbstop does not know how to connect to the server to tell it to shutdown.

The dbstop command line options let you control whether a server is stopped, even if there are active connections.

server-name Name of the server to stop. If you supply a server name, do not supply connection parameters as well.

-c When stopping a network server, you must supply a connection string with a user ID that has permissions to stop the server. By default, DBA permission is required on the network server, and all users can shut down a personal server; you can use the -gk server command-line option to change the default behavior.

For a description of the connection parameters, see Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

If there are active connections, dbstop prompts whether you want to shut down the server. If you specify `unconditional=true` on the command line, the server shuts down without prompting, even if there are active connections.

-d Do not stop the database server. Instead, only stop the database specified in the connection string.

-o *filename* Write output messages to the named file.

-q Run in quiet mode—messages are not displayed.

-x Do not stop the server if there are still active connections to the server. Including this option prevents dbstop from prompting for confirmation if there are active connections.

-y Stop the server even if there are still active connections to the server.

Examples

To stop a database named iqdemo on the server myserver, enter:

```
dbstop -c "uid=DBA;pwd=sql;eng=myserver;dbn=iqdemo"
```

To stop a server myserver, regardless of the database running, enter:

```
dbstop -c
"uid=DBA;pwd=sql;eng=myserver;dbn=utility_db"
```

See also

Chapter 2, “Running Sybase IQ” in *System Administration Guide: Volume 1*

stop_iq (UNIX and Linux only)

On UNIX and Linux platforms, you can also stop the database server using the stop_iq command as well as using dbstop. For full details on when to use stop_iq, see the section “Stopping the database server” in Chapter 2, “Running Sybase IQ” in the *System Administration Guide: Volume 1*.

Syntax

stop_iq [**-agent** | **-cleanup**] [**-stop** [**one** | **all**]] [**-user** <*user_name*>] [**-version** [**12** | **15** | **all**]>] [**-wait** <*seconds*>]

Switches

Table 1-4: Switches for stop_iq utility

Parameter	Purpose
-agent	Stops the IQ Agent on UNIX or Linux systems
-cleanup	Removes the orphan IQ process on Linux
-stop [one | all]	Removes user interaction with stop_iq. Assumes a yes response to all questions.
-user	Specify a user to stop server processes.
-version	Specify the version of Sybase IQ the switch applies to.
-wait	Specify the time to wait for the server to shut down before timeout expires.

Usage

stop_iq is an interactive utility that lists all of the servers owned by other users, then lists the servers you own, then asks you if you want to stop your server. If you respond Y (yes), stop_iq shuts down your server and closes all user connections to your server. If you respond N (no), stop_iq returns to the operating system command prompt, and your server continues to run.

When you issue stop_iq, you see this message:

```
"Please note that 'stop_iq' will shutdown a server
completely without regard for users connections or
load processes status. For a finer level of detail
the utility 'dbstop' has the options to control
whether a server is stopped based on active
connections."
```

Normally, do not shut down a server while it is still connected to one or more clients. If you try this, you get a warning that any uncommitted transactions will be lost. Disconnect or close all the clients and try again.

The -agent option shuts down the IQ Agent on UNIX and Linux.

The -version switch specifies the version of Sybase IQ that is being used.

The -user parameter performs two functions.

- If, due to truncation or substitution, stop_iq cannot find the server or agent owned by the current user, the system manager can specify the name and id found in the process table in the -user argument to shut down the server/agent with the stop_iq utility.

- A user with root privileges can shut down another user’s server or agent without having to log in as that user. The stop_iq utility has no superuser (su) or root powers, so a nonprivileged user cannot shut down a server owned by another user.

Stopping servers in cron or at jobs

To use stop_iq in a cron or at job, use the appropriate -stop option:

```
stop_iq -stop one
```

Setting -stop one shuts down a single server, when exactly one running server was started by the user ID that starts the cron or at job. This prevents the wrong server from being shut down if several are running.

```
stop_iq -stop all
```

Setting -stop all shuts down all servers that were started by the user ID that starts the server.

You can specify both options on the same command, for example:

```
stop_iq -agent -stop all
```

Note In a cron statement, you must specify the complete path name to the stop_iq executable.

Example

The following example illustrates how to use stop_iq interactively to list all the servers that are running, and then shut down one server.

```
% stop_iq
Checking system ...
The following 1 server(s) are owned by 'TEST'

##     Owner    PID   Started  CPU_Time   Additional Information
 -- ---------  -----  --------  --------  -----------------------
1:   TEST     22399  08:56:39      1:43  SVR:QA_sun7qa DB:iqdemo PORT:8888
/sun7qa1/users/QA/090513/IQ-15_1/bin64/iqsrv15 @iqdemo.cfg iqdemo.db -ti
4400
--
   Please note that ’stop_iq’ will shutdown a server completely
   without regard for users connections or load processes status.
   For a finer level of detail the utility ’dbstop’ has the options
   to control whether a server is stopped based on active connections.

Do you want to stop the server displayed above <Y/N>? Y

Shutting down server (22399) ...
Checkpointing server (22399) ...
Server shutdown.
```

See also Chapter 2, “Running Sybase IQ” in *System Administration Guide: Volume 1*

Troubleshooting the database server

If you have problems with the operation of the server, including startup, shutdown, unresponsiveness, and abnormal termination, refer to Chapter 14, “Troubleshooting Hints” in the *System Administration Guide: Volume 1* for troubleshooting information.

CHAPTER 2 Using Interactive SQL (dbisql)

About this chapter

This chapter describes Interactive SQL (dbisql). In addition to syntax and parameters, the chapter includes a tutorial on starting Sybase IQ, opening the demo database, using the command window, cancelling commands, and stopping Interactive SQL.

Contents

Introduction to Interactive SQL

Interactive SQL (dbisql) is a utility for sending SQL statements to the database server. You can use this utility to:

- Browse the information in a database.
- Try out SQL statements that you plan to include in an application.
- Load data into a database and carry out other administrative tasks.

In addition, Interactive SQL can run command files. You can build repeatable scripts to run against a database, then use dbisql to execute these scripts. See "Saving, loading, and running command files" on page 64.

dbisql and dbisqlc

dbisql is a Java-based application. In this book, Interactive SQL always refers to Interactive SQL Java . dbisql has several advantages over the older Windows-based Interactive SQL Classic application (dbisqlc):

- Find server capability – the Connect window features a Find button beside the Server Name box on the Database tab. This allows you to browse and select the server name. If your server is remote, select Search network for database servers, which, by default, is unselected.
- Mouse buttons on UNIX – you can select items from the dbisql menus using the mouse buttons.
- Cut and paste capability – available from both the main menu Edit item and a menu that appears when you select and right-click text in the SQL Statements window.

To start dbisql, type dbisql at a UNIX command prompt, or choose Programs > Sybase > Sybase IQ 15.1 > Interactive SQL Java on Windows.

For backward compatibility, Sybase IQ still includes Interactive SQL Classic (dbisqlc) if you need a feature not yet implemented in Interactive SQL Java. See "The Interactive SQL Classic (dbisqlc) utility" on page 78 for dbisqlc syntax.

To start Interactive SQL Classic, enter dbisqlc at a UNIX command prompt, or choose Programs > Sybase > Sybase IQ 15.1 > Interactive SQL Classic on Windows.

The Interactive SQL (dbisql) utility

dbisql provides an interactive environment for database browsing and for sending SQL statements to the database server.

dbisql allows you to type SQL commands or run command files. dbisql also provides feedback about the number of rows affected, the time required for each command, the execution plan of queries, and any error messages.

For dbisql to function correctly, $HOME must exist and must be writable by the user.

To start dbisql, see “Opening Interactive SQL using the dbisql command-line utility” on page 44 and “Starting dbisql” on page 47.

Opening Interactive SQL using the dbisql command-line utility

Syntax **dbisql** [*options*] [*dbisql-command* | *command-file*]

Parameters

Table 2-1: dbisql options

Option	Description
-c *"keyword=value; ..."*	Supply database connection parameters.
-codepage *codepage*	Specify a code page to use when reading or writing files.
-d *delimiter*	Use the given string as the command delimiter.
-d1	Print statements as they are executed [command-prompt mode only].
-datasource *dsn-name*	Supply an ODBC data source.
-f *filename*	Open (without running) the file called *filename*.
-host *hostname*	Specify the host name or IP address of the machine running a database server.
-jConnect	Use jConnect™ to connect to the database.
-nogui	Run in command-prompt mode.
-ODBC	Use the iAnywhere JDBC driver to connect to the database.
-port *portnumber*	Look on the specified port number for the database server.
-q	Run in quiet mode — No windows or messages.
-x	Run in syntax check mode — No commands executed.

Usage

Exit codes are 0 (success) or non-zero (failure).

Interactive SQL requires that you set the QUOTED_IDENTIFIER database option ON since a number of database functions, including some statements, rely on this setting to function. Interactive SQL automatically sets this option ON when connecting to a database.

Interactive SQL does not accept @filename parameters.

dbisql-command | command-file If *dbisql-command* is specified, dbisql executes the command. You can also specify a command file name. If no *dbisql-command* is specified, dbisql enters interactive mode, where you can type a command into a command window. From the Windows Start menu, you can enter interactive mode by selecting Programs > Sybase > Sybase IQ 15.1 > Interactive SQL Java.

-c "keyword=value; ..." Specify connection parameters. See Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1* for a description of the connection parameters. If you do not specify this option, the environment variable SQLCONNECT is used. If Interactive SQL cannot connect, you see a dialog box where you can enter the connection parameters.

Note Sybase recommends that, to avoid ambiguity, you always specify connection parameters for dbisql instead of relying on defaults. If you start more than one database on a server, for example, specify the database name, and in a network with subnets, specify the communications protocol parameter with host number. See Chapter 3, “Sybase IQ Connections” in the *System Administration Guide: Volume 1*.

-codepage Specify the code page to use when reading or writing files. The default code page is for the platform you are running on.

For example, on an English Windows machine, Interactive SQL uses the 1252 (ANSI) code page. If you want Interactive SQL to read files created using the 297 (IBM France) code page, specify:

```
-codepage 297
```

-d Specify a command delimiter. Quotation marks around the delimiter are optional, except when the command shell itself interprets the delimiter in some special way.

Command delimiters are used for all connections in that Interactive SQL session, regardless of the setting stored in the database (for the user, or the PUBLIC setting).

-d1 (The final character is a number 1, not a lower case L.) Interactive SQL echoes all statements it executes to the Command window (STDOUT). You may find this feedback useful if you are debugging SQL scripts, or when Interactive SQL is processing a long SQL script.

-datasource Specify an ODBC data source to connect to. You do not need to be using the iAnywhere JDBC driver to use this option. However, if the data source to which you are connecting is not configured to use TCP/IP, you must use the iAnywhere JDBC driver to connect. By default, Sybase IQ data sources are configured to use TCP/IP.

-f *filename* Open (but do not run) the file called *filename*. You must enclose the file name in quotes if the file name contains a blank; otherwise, quotes are optional. If the file does not exist, or if it is a directory instead of a file, Interactive SQL prints an error message to the console and quits. If the file name does not include a full drive and path specification, the file is assumed to be relative to the current directory.

-host Specify the host name or IP address of the computer on which the database server is running. You can use the name localhost to represent the current machine.

-jConnect Use the jConnect JDBC driver to connect to the database. For most circumstances, Sybase recommends this driver.

-nogui Run Interactive SQL in a command-prompt mode, with no windowed user interface. This is useful for batch operations. If you specify either *dbisql-command* or *command-file*, then -nogui is assumed.

In -nogui mode, Interactive SQL sets the program exit code to indicate success or failure. On Windows operating systems, the environment variable ERRORLEVEL is set to the program exit code.

Table 2-2: dbisql program exit codes

Program exit code	Description
0	Success.
1	General failure. At some point, a SQL or Interactive SQL statement did not execute successfully and the user chose to stop executing SQL statements. Alternatively, Interactive SQL noted an internal error.
5	User terminated Interactive SQL. When an error occurs during execution, the user is prompted to ignore the error, stop, or exit Interactive SQL. If the user opts to exit, the program returns code 5.
9	Unable to connect.
255	Bad command line. The command line contained incomplete or invalid switches.

-ODBC Connect using the iAnywhere JDBC driver. The iAnywhere JDBC driver was formerly called the JDBC-ODBC bridge.

-onerror Controls what happens if an error is encountered while reading statements from a command file.

-port Specify the port number on which the database server is running. The default port number for Sybase IQ is 2638.

-q Do not display output messages. This option is useful only if you start Interactive SQL with a command or command file. Specifying this option does not suppress error messages.

-x Scan commands but do not execute them. You may find this option useful for checking long command files for syntax errors.

Examples

The examples below show how to use dbisql at the system command prompt level. The rest of the dbisql sections in this chapter discuss how to use dbisql interactively.

- The following command, entered at a system prompt, runs the command file *mycom.sql* against the current default server, using the user ID dba and the password sql. If there is an error in the command file, the process terminates.

  ```
  dbisql -c "uid=DBA;pwd=sql" -onerror exit mycom.sql
  ```

- The following command, when entered on a single line at a command prompt, adds a user to the current default database:

  ```
  dbisql -c "uid=DBA;pwd=sql" grant connect to joe
  identified by passwd
  ```

See also

For detailed descriptions of SQL statements and dbisql commands, see Chapter 1, “SQL Statements” in *Reference: Statements and Options*.

Starting dbisql

You can start Interactive SQL in two ways: from Sybase Central or in standalone mode. The way that you start Interactive SQL on its own depends on your operating system.

For detailed information on connecting to databases, see Chapter 3, “Running and Connecting to Servers” in *Introduction to Sybase IQ*.

❖ Starting Interactive SQL from Sybase Central

1. In the left pane, select the Sybase IQ plug-in.
2. Choose Tools > Sybase IQ 15 > Open Interactive SQL.
3. In the Connect window, supply parameters as described in “Connection parameters” on page 49.

Tip
You can also access Interactive SQL from within Sybase Central by :

- Selecting a database in the left pane, and choosing File > Open Interactive SQL.
- Right-clicking a database, and choosing Open Interactive SQL.
- Right-clicking a stored procedure, and choosing Execute from Interactive SQL. Interactive SQL opens with a CALL to the procedure in the SQL Statements pane and executes the stored procedure.

❖ Starting Interactive SQL in standalone mode on Windows

1. Select Start > Programs > Sybase > Sybase IQ 15.1 > Interactive SQL Java.
2. In the Connect window, supply parameters as described in “Connection parameters” on page 49.

❖ Starting Interactive SQL in standalone mode on UNIX

1. If the sample database server is not already running, change to the *demo* directory and execute the following command:

   ```
   % start_iq -n servername iqdemo.db
   ```

   For *servername*, substitute a unique server name, for example, your system name, database name, and port number, separated by an underscore.

   For example:

   ```
   cd $SYBASE/IQ-15_1/demo
   start_iq -n fiona_iqdemo_1870 iqdemo.db
   ```

   Note If you specify -n *servername* without a *dbname*, the server connects to the default database on the current server. If you specify -n *dbname* without a *servername*, Sybase IQ connects to that database on the current server.

2. To start dbisql, type the following command at a system prompt: dbisql

Once the database server is running, you can also start dbisql by typing `dbisql` at the Command Prompt or by right-clicking the server name in Sybase Central and choosing Interactive SQL from the pop-up menu

Connection parameters

When no database is connected, Sybase IQ displays the Connect window to request connection parameters. Information you enter in the Connect window is not preserved between sessions.

The connection parameters you specify in the Connect window are dependent on the number of databases running on the database server. To connect to a single database, you complete the User ID and Password fields. If there are multiple databases running on the database server, you must specify additional connection parameters such as the server or database name.

Enter connection parameters in the Connect window using one of these methods:

- Use the Connect Assistant. See "Supplying connection parameters using the Connect Assistant" on page 50.
- Manual enter connection parameters using the following tabs:
  - Identification tab– see "Supplying connection parameters: Identification tab" on page 50.
  - Database tab – see "Supplying connection parameters: Database tab" on page 52.
  - Network tab – see "Supplying connection parameters: Network tab" on page 54.
  - Advanced tab – see "Supplying connection parameters: Advanced tab" on page 56.

You can bypass the Connect window by supplying the connection parameters on the command line with the -c option as shown in "Examples" on page 47.

You can also click Cancel to start Interactive SQL with no database connected. You can then enter a CONNECT command in the SQL Statements window. For example, to connect to the sample database on a local server, you can use:

```
CONNECT USING 'UID=DBA;PWD=sql'
```

If the Connect window or an error message about missing information appears, you may need to enter the -host and -port or other missing information in the Advanced tab. If your database is on a remote server, enter the -host and -port parameters on separate lines, as in:

```
-host fiona
-port 1870
```

For a complete list of connection parameters, see Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

After connecting to the database, the Interactive SQL utility appears.

Supplying connection parameters using the Connect Assistant

The Connect Assistant is a wizard that helps you connect to a database.

The Connect window has a Connect Assistant to help you connect to a database. To display or hide the Connect Assistant, click the arrow in the top right corner of the window.

To run the Connect Assistant, click the Next button in the Connect Assistant area of the Connect window and follow the on-screen prompts.

Note If you are connecting using an ODBC source, do not use the Connect Assistant. Close the Connect Assistant and click the Identification tab. See “Supplying connection parameters: Identification tab” on page 50.

Supplying connection parameters: Identification tab

When no database is connected, Sybase IQ displays the Identification tab, requesting connection details.

❖ **Specifying identification information**

1 If you are not using integrated login on Windows, select Supply user ID and password.

   - In the User ID field, enter dba for the iqdemo database. The user ID is not case-sensitive.
   - In the Password field, enter sql for the iqdemo database. The password is case-sensitive.

2 If applicable, select Use integrated login instead of supplying a user ID and password.

3 Choose one of these default connection profiles:

- None
- ODBC Data Source name
- ODBC Data Source file

4 Click the Database tab when finished.

See “Connect window: Identification tab components” on page 51 for information on the fields, buttons, and options that appear on the Identification tab.

Connect window: Identification tab components

The Identification tab has the following components:

Supply user ID and password Select this option to specify a user ID and password for the connection.

- User ID – type a user ID for the connection. The default user ID is dba. The user ID must have permissions to connect to the database.
- Password – type a password for the connection. The default password for the iqdemo database is sql when you connect with the default user, dba. Passwords are case-sensitive.

Use integrated login Select this option to connect to the database using an integrated login on Windows. To use this option, the DBA must define an integrated login for you. See “Using an integrated login” in Chapter 3, “Sybase IQ Connections” in the *System Administration Guide: Volume 1*.

None Select this option if you are not using an ODBC data source for the connection.

ODBC data source name Select this option to choose a data source, which is a stored set of connection parameters, for connecting to your database. This field is equivalent to the DSN connection parameter, which references a data source in the registry. You can view a list of data sources by clicking Browse or select a recently used ODBC data source from a list.

- Open the ODBC Data Source Administrator button – Open the ODBC Administrator window, where you can select an ODBC data source from the list of available data sources. You may also choose to create a new data source or configure an existing data source to use for the connection.

Tip
ODBC data sources allow you to store connection information. If you have connection information stored in an ODBC data source, do not repeat this information in the Connect window. For example, if the ODBC data source already contains a user ID, then do not supply the user ID in the User ID field of this Connect window.

Any information you enter in the Connect window, such as the user ID or database name, takes precedence over the parameters stored in an ODBC data source or an ODBC data source file.

ODBC data source file Select this option to choose a data source file for the connection. You can search for the file by clicking Browse or selecting a recently used ODBC data source file from a list. The ODBC data source file is often used for UNIX systems.

Supplying connection parameters: Database tab

Use the Database tab to identify the database server and database.

❖ Specifying database information

1. Enter the server name you used when starting the server.
2. To connect to a local database server that is not currently running, and to set your own start parameters rather than using the defaults, enter a start line.
3. If there is more than one database running on the database server, enter the database file name, without any suffix.
4. If the database you are connecting to is not currently running on a database server, specify the database file.
5. Enter the encryption key to the database server. The Encryption key field is enabled only after you fill in the Database File field.
6. Select Start database automatically to start the database specified in the Database file field before you connect to it.
7. Select Stop database after last disconnect to automatically shut down the database after the last user disconnects.

8 Click OK if you are ready to connect, or click the Network tab to specify any required network options.

See "Connect window: Database tab components" on page 53 for information on the fields, buttons, and options that appear on the Database tab.

Connect window: Database tab components

The database tab has the following components.

Server name Type the name of the database you are connecting to. For remote servers, specify the server as *host name*:*port number*.

You can choose a recently-used database server name or click Find to search for a server. When you click Find, you see a list of running local personal servers and network servers. Select a database server from the list and click OK. The database server name appears on the Database tab in the Server Name field.

Start line A start line is a command to start a personal or a network database server on your computer. Enter a start line only to connect to a local database server that is not currently running and to set your own start parameters. You must enter the full path of the database server. Alternatively, you can choose a recently used start line from the dropdown list

Database name Each database running on a database server is identified by a database name. Type the name of the database you are connecting to. You need a database name only if there is more than one database running on the database server. If the database is not already running on the server, you should specify the database file instead. You can also select a recently used database, or click Browse to locate the database file.

Note If you specify both the database name and database file when trying to connect to a database that is already running, the database file is ignored.

Database file Specify the database file when the database you want to connect to is not currently running on a database server. Sybase recommends that you type the full path and name of the database file. Otherwise, the path of the file is relative to the working directory of the database server. You can also choose a recently-used database file from the dropdown list or click Browse to search for the database file.

- Encryption key – If the database file is encrypted, you must supply a key to the database server every time the database server starts the database. The Encryption key field is enabled only after you fill in the Database File field. You can also supply encryption options in the Start Line field.

Start database automatically Select this option to start the database specified in the Database file field before you connect to it. To ensure that you connect only to a running database, unselect this option.

Stop database after last disconnect Select this option to automatically shut down the database after the last user disconnects.

Supplying connection parameters: Network tab

Use the Network tab to specify network options.

❖ Specifying network information

1. If you are using same-computer communications, select Shared Memory protocol.
2. If you are connecting to a server on another computer, select TCP/IP protocol:
   - Host – specify the computer name on which the database server is running.

     Click Ping to test that a computer with the given host name can be found on your network.
   - Ports – specify the port the database server is using.
   - Other – specify any other network protocol options. Click Edit to open the TCP Properties window.
3. Select one of these security options: none, simple, or tls.
4. Click OK if you are ready to connect, or click the Advanced tab to specify any advanced connection options.

See “Connect window: Network tab components” on page 55 for information on the fields, buttons, and options that appear on the Network tab.

Connect window: Network tab components

The Network tab has the following components:

Shared Memory This protocol is for same-computer communications, and always remains available. It is available on all platforms.

TCP/IP Choose this protocol if you are connecting to a server running on another computer. This protocol is supported on all platforms.

- Host – specify the computer name on which the database server is running.
- Ping – click to test that a computer with the given host name can be found on your network.
- Ports(s) – if your database server is not using the default port, you must specify the port it is using.
- Other – specify any other network protocol options.
- Edit – open the TCP Properties window that provides a list of TCP properties that you can specify.

Security Choose one of the following: none, simple, or tls. When you choose tls for the Security option, the encryption connection parameters table is enabled:

- certificate_company – the application only accepts server certificates when the Organization field on the certificate matches this value.
- certificate_name – the application only accepts server certificates when the Common Name field on the certificate matches this value.
- certificate_unit – the application only accepts server certificates when the Organization Unit field on the certificate matches this value.
- fips – choose whether to use FIPS-approved encryption implementations for TLS encryption and end-to-end encryption.
- tls_type – specify either ecc or rsa as the encryption cipher to use for synchronization.
- trusted_certificates – specify a file containing a list of trusted root certificates used for secure synchronization.

Supplying connection parameters: Advanced tab

Use the Advanced tab to set advanced connection parameters. Connection parameters set in the Advanced tab are superseded by parameters set in the other tabs in the Connect window. For example, if you enter the user ID dba on the Identification tab, and set the connection parameter "UID=bsmith" on the Advanced tab, Sybase IQ attempts a connection with the user ID dba.

❖ **Specifying advanced network connection parameters**

1. In the Value column, click the advanced network connection parameter value you want to modify. The parameter description appears in the bottom of the Connect window.
2. Modify the parameter value.
3. Modify any other advanced network connection parameters. Sybase IQ remembers your changes when you click another cell in the Value column.
4. Click OK when you are ready to connect.

Connect window tools

Click Tools at the bottom of the Connect window to access these options:

Test connection This tool tests whether the information provided results in a proper connection. This tool is available only when you are connecting to a SQL Anywhere database.

Copy connection string to clipboard This tool creates a connection string from the options you specified in the Connect window ,and copies the string into your clipboard.

Save as ODBC Data Source This tool creates an ODBC data source based on the information provided.

Main dbisql window description

Interactive SQL has the following panes:

- SQL Statements provides a place for you to type SQL statements. If the code you type exceeds the size of the pane, scroll bars automatically appear.
- Messages displays information about execution. You can select message options from the Tools > Options dialog, including whether to display messages in a separate pane or as a tab on the Results pane (the default).
- Results displays the results of commands that you execute. For example, if you use SQL statements to retrieve specific data, the Results pane shows the columns and rows that match the search criteria. If the information exceeds the size of the pane, scroll bars automatically appear.

Note When you execute queries, Interactive SQL also displays a SQL Anywhere query plan in the Plan pane. You can ignore these plans for queries against the IQ store and use the IQ query plan in the *.iqmsg* file or the HTML query plan.

Authorized users can also display, save, and print query plans from dbisql instead of accessing the *.iqmsg* file or query plan files on the server. Using the SQL functions GRAPHICAL_PLAN and HTML_PLAN, you can retrieve IQ query plans in XML and HTML format, respectively, as a string result set.

See “Planning queries” and “Using query plans” in *Performance and Tuning Guide*.

Opening multiple windows

You can open multiple Interactive SQL windows, each corresponding to a separate database connection. Use multiple windows to connect simultaneously to two or more different databases on different servers, or to open concurrent connections to a single database.

❖ **To open a new Interactive SQL window:**

1. Choose Window > New Window.
2. In the resulting Connect window, enter connection options, and click OK to connect.

You can also connect to or disconnect from a database with the Connect and Disconnect commands in the SQL menu, or by executing a CONNECT or DISCONNECT statement.

Keyboard shortcuts

Interactive SQL provides the keyboard shortcuts listed in Table 2-3.

Table 2-3: dbisql keyboard shortcuts

Function key	Description
Alt+F4	Exits Interactive SQL.
Alt+Left cursor	Displays the previous SQL statement in the history list.
Alt+Right cursor	Displays the next SQL statement in the history list.
Ctrl+Break	Interrupts the SQL statement that is being executed.
Ctrl+C	Copies the selected rows and column headings to the clipboard in the Results pane.
	In the SQL Statements pane, copies the selected text to the clipboard.
Ctrl+End	Moves to the bottom of the current pane.
Ctrl+H	Displays the history of your executed SQL statements.
Ctrl+Home	Moves to the top of the current pane.
Ctrl+N	Clears the contents of the Interactive SQL window.
Ctrl+P	Prints the contents of the SQL Statements pane.
Ctrl+Q	Displays the Query Editor.
	The Query Editor helps you build SQL queries. When you have finished building your query, click OK to export it back into the SQL Statements pane.
Ctrl+S	Saves the contents of the SQL Statements pane.
Esc	Clears the SQL Statements pane.
F1	Opens Help.
F2	Edits the selected value in the result set. You can use the Tab key to move from column to column within the row.
F5	Executes all text in the SQL Statements pane.
F7	Displays the Lookup Table Name dialog.
F8	Displays the Lookup Procedure Name dialog.
F9	Executes the text that is selected in the SQL Statements pane.
	If no text is selected, all of the statements are executed.
Pgdn	Moves a page down in the current pane.
Pgup	Moves a page up in the current pane.

The keyboard shortcuts in Table 2-4 are available when the SQL Statements pane has the focus:

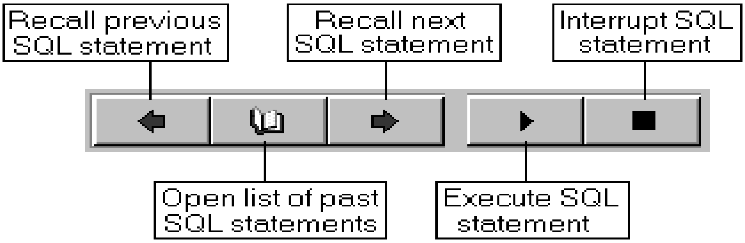
Table 2-4: dbisql keyboard shortcuts for SQL Statements pane

Function key	Description
Ctrl+]	Moves the cursor to the matching parenthesis, braces, brackets, and angle brackets.
Ctrl+Backspace	Deletes the word to the left of the cursor.
Ctrl+Del	Deletes the word to the right of the cursor.
Ctrl+G	Opens the Go To dialog where you can specify the line you want to go to.
Ctrl+L	Deletes the current line from the SQL Statements pane and puts the line onto the clipboard.
Ctrl+Shift+]	Extends the selection to the matching brace. Brace matching matches parentheses, braces, brackets, and angle brackets.
Ctrl+Shift+L	Deletes the current line.
Ctrl+Shift+U	Changes the selection to uppercase characters.
Ctrl+U	Changes the selection to lowercase characters.
F3	Finds the next occurrence of the selected text.
Home	Moves the cursor to the start of the current line or to the first word on the current line.
Shift+F3	Finds the previous occurrence of the selected text.
Shift+Home	Extends the selection to the start of the text on the current line.

Using the Interactive SQL toolbar

The Interactive SQL toolbar (shown in Figure 2-1) appears at the top of the Interactive SQL window.

Figure 2-1: Interactive SQL toolbar


Recall previous SQL statement
Recall next SQL statement
Interrupt SQL statement
Open list of past SQL statements
Execute SQL statement

With the buttons on this toolbar, you can:

- Recall the executed SQL statement immediately before your current position in the history list.
- View a list of up to 50 previously executed SQL statements.
- Recall the executed SQL statement immediately after your current position in the history list.
- Execute the SQL statement currently showing in the SQL Statements pane.
- Interrupt the execution of the current SQL statement.

Hover your cursor over each button to see a description.

Using Interactive SQL to display data

You can use Interactive SQL to browse the information in databases. This section shows how to query the information in the demo database.

You can display database information using a SELECT statement in Interactive SQL. The following example shows the command to type in the SQL Statements pane. Once you have typed the command, click the Execute All SQL Statement(s) button on the toolbar to carry out the command.

After you execute the statement, the data (called a result set) appears in the Results pane. You can use the scroll bars to see areas of the table that are outside your current view of the pane.

❖ **Listing the columns and rows in the Employees table:**

1 Start Interactive SQL and connect to the demo database.

2 In the SQL Statements pane, enter:

```
SELECT *
FROM Employees
```

3 Select Execute SQL Statement from the toolbar.

EmployeeID	**ManagerID**	**Surname**	**GivenName**	**...**
102	501	Whitney	Fran	...
105	501	Cobb	Matthew	...
129	902	Chin	Philip	...
148	1293	Jordan	Julie	...

EmployeeID	ManagerID	Surname	GivenName	...
160	501	Breault	Robert	...
...				

For more information on SELECT statements, see Chapter 1, “Selecting Data from Database Tables” in the *Performance and Tuning Guide*.

Working with commands in Interactive SQL

This section describes general tasks for working with commands in Interactive SQL.

All SQL statements can be entered as commands in the top pane of the Interactive SQL viewer. When you are finished typing, execute the statement to run it.

- Enter a command in the SQL Statements pane.
- To execute a command, click Execute SQL Statement, select SQL > Execute, or press F5.
- To clear the SQL Statements pane, Choose Edit > Clear SQL or press ESCAPE.

You can execute only a selection of text in the SQL Statements pane by pressing F9.

Other messages about an executed statement, such as execution times, can appear in the Messages pane. Use Tools > Options > Messages to change the appearance and content of message information.

Combining multiple statements

You can enter multiple statements into Interactive SQL; use a semicolon (;) to separate each statement.

❖ **Entering multiple statements in SQL Statements pane**

1 Enter these commands into the SQL Statements pane.

```
UPDATE Employees
SET DepartmentID = 400,
    ManagerID = 1576
WHERE EmployeeID = 467;

UPDATE Employees
SET DepartmentID = 400,
    ManagerID = 1576
WHERE EmployeeID = 195;

SELECT *
FROM Employees
WHERE EmployeeID IN ( 195, 467 );
```

2 On the toolbar, click Execute SQL Statement. All three statements are executed. After execution, the commands remain in the SQL Statements pane. To clear this pane, press the Esc key.

Looking up tables, columns, and procedures

While you are entering commands in Interactive SQL, you can look up the names of tables, columns, or procedures stored in the current database and insert them at your cursor position.

❖ **Looking up the names of tables in the database**

1 Choose Tools > Lookup Table Name.

2 Find and select the table.

3 Click OK to insert the table name into the SQL Statements pane.

❖ **Looking up column names in the database**

1 Choose Tools > Lookup Table Name.

2 Find and select the table containing the column.

3 Click Show Columns.

4 Select the column and click OK to insert the column name into the SQL Statements pane.

❖ **Looking up the names of procedures in the database**

1 Choose Tools > Lookup Procedure Name.

2 Find and select the procedure.

3 Click OK to insert the procedure name into the SQL Statements pane.

In the tables and procedures lookup dialogs, you can enter the first few characters of the table or procedure you are looking for. The dialog then narrows the list to include only those items that start with the text you entered.

You can use the standard SQL wildcard `%` to mean “match anything.” For example, to list only those tables ending in order, type `%order` into the lookup field. To list all items containing the word java, type `%java%`. Clearing the search criteria from the look up field resets the filtering to display all the items.

Recalling commands

When you execute a command, Interactive SQL automatically saves the command in a history list, of up to 50 of the most recent commands, that lasts for the duration of the current session.

You can view the entire list of commands in the Command History dialog. To access the Command History dialog, press CTRL+H, or click the book icon in the toolbar.

The most recent commands appear at the bottom of the list. To recall a command, highlight the command and click OK. The command appears in the SQL Statements pane.

In Windows, you can also recall commands without the Command History dialog. Use the arrows in the toolbar to scroll back and forward through your commands, or press ALT + RIGHT ARROW and ALT + LEFT ARROW. (The ALT + RIGHT ARROW and ALT + LEFT ARROW key combinations do not scroll through commands in UNIX.)

You can also save commands in text files, so that you can use them in a subsequent Interactive SQL session as described in the next section.

Saving, loading, and running command files

You can save the commands in the SQL Statements pane so that they are available for future Interactive SQL sessions. The file in which you save them is called a command file, or a script.

Note ODBC applications, including Interactive SQL applications, automatically set certain database options to values mandated by the ODBC specification. These settings overwrite those set in the LOGIN_PROCEDURE database option. For details and a workaround, see “LOGIN_PROCEDURE option,” in Chapter 2, “Database Options” of *Reference: Statements and Options*.

When you begin a new session, you can load the contents of a command file into the SQL Statements pane, or you can run the contents immediately.

❖ Saving the commands in the SQL Statements pane

1 Choose File > Save As.

2 In the Save dialog, specify a location, name and format for the file. Click Save when finished.

❖ Loading commands from a file into the SQL Statements pane

1 Choose File > Open.

2 In the Open dialog, find and select the file. Click Open when finished.

❖ Running a command file immediately

1 Choose File > Run Script.

2 In the Open dialog, find and select the file. Click Open when finished.

You can supply a command file as a command-line argument for Interactive SQL.

The Run Script menu item is the functional equivalent of a READ statement. For example, in the SQL Statements pane, you can also run a command file by typing:

```
READ 'c:\filename.sql'
```

where *c:\filename.sql* is the path, name, and extension of the file. You can enclose the string in single quotation marks (as shown) if you want, but they are required only if the path contains embedded spaces.

Logging commands

With the Interactive SQL logging feature, you can record commands as you execute them. Interactive SQL continues to record until you stop the logging process, or until you end the current session. The recorded commands are stored in a log file.

❖ **Logging Interactive SQL commands:**

1. Choose SQL > Start Logging.
2. In the Save dialog, specify a location and name for the log file.
3. Click Save when finished.
4. Choose SQL > Stop Logging.

You can also start and stop logging by typing in the SQL Statements pane. To start logging, type and execute START LOGGING *'c:\filename.sql'*, where *c:\filename.sql* is the path, name, and extension of the log file. A log file must have the *.sql* extension. You only need to include the single quotation marks if the path contains embedded spaces. To stop logging, type and execute STOP LOGGING.

Once you start logging, all commands that you try to execute are logged, including ones that do not execute properly.

Canceling an Interactive SQL command

An interrupt or stop operation stops current processing and prompts for the next command or action.

❖ **To interrupt an Interactive SQL command:**

- To interrupt an Interactive SQL command, select Interrupt the SQL Statement from the toolbar, or select SQL > Stop.

If a command file is being processed, select “Stop,” “Continue,” or “Exit.”

Reported errors

When an abort is detected, one of three different errors will be reported depending upon when the abort is detected.

1. If the abort is detected when Interactive SQL is processing the request (as opposed to the database engine), this message appears:

```
ISQL command terminated by user
```

Interactive SQL immediately stops processing and the database transaction is left alone.

2 If the cancel is detected by the database engine while processing a standard data manipulation command (SELECT, INSERT, and DELETE) and the engine is not running in bulk operations mode, this message appears:

```
    Statement interrupted by user
```

The effects of the current command are undone, but the rest of the transaction is left intact.

3 If the abort is detected while the database engine is processing a data definition command (CREATE, DROP, ALTER, and so on) this message appears:

```
Terminated by user -- transaction rolled back
```

Since data definition commands all perform a COMMIT automatically before the command starts, the effect of the ROLLBACK is simply to cancel the current command.

This message also occurs when the database engine is running in bulk operations mode executing a command that modifies the database (INSERT, UPDATE, or DELETE). In this case, ROLLBACK cancels not only the current command, but everything that has been done since the last COMMIT. In some cases, it may take a considerable amount of time for the database engine to perform the automatic ROLLBACK.

Configuring Interactive SQL

You can configure different Interactive SQL viewer options in the Options window. This window provides settings for commands, viewer appearance, import and export features and messages.

You can set these options either using the Interactive SQL GUI, or by using the SET OPTION statement. See SET OPTION statement [DBISQL] in Chapter 1, “SQL Statements” in *Reference: Statements and Options*.

❖ Accessing the Options window

1 Choose Tools > Options.

2 Click an option in the left pane.

The options of the Options window are described in the following sections.

Note Some Interactive SQL options are for use with SQL Anywhere only, and are not discussed here.

Options window: General

The General page of the Options window has these components:

- **Command files** These options control Interactive SQL behavior while it is running command files.
  - **When an error occurs** Select one of the following options to control how Interactive SQL responds when it encounters an error while executing statements:
    - Stop – Interactive SQL stops executing statements.
    - Prompt – Interactive SQL prompts the user whether to continue. This is the default.
    - Continue – the error is ignored and Interactive SQL continues executing statements.
    - Exit – interactive SQL shuts down.
    - Notify and continue – the error is reported, and the user is prompted to press Enter or click OK to continue.
    - Notify and stop – the error is reported and the user is prompted to press Enter or click OK to stop executing statements.
    - Notify and exit – the error is reported and the user is prompted to press Enter or click OK to shut down Interactive SQL.

    Note When you are executing a *.sql* file, the values Stop and Exit are equivalent. If you specify either of these values, Interactive SQL shuts down.

  - **Echo command files to log** Select this option to log executed SQL statements to the log file. You may find this option to be useful when you use the READ statement to execute an Interactive SQL command file or when you run a command file in Interactive SQL by choosing File > Run Script. Logging must be turned on for this option to take effect. By default, command files are copied to the log.

- **Folders** Use these options to specify which directory Interactive SQL uses first when browsing for files.
  - **Last folder used** If you select this option, the initial browser directory is the one where you last used the file browser. This is the default.
  - **Current folder** If you select this option, the initial directory used by the browser is the current folder, as defined by the operating system.

Options window: Import/Export

The Import/Export page of the Options window has these components:

- **Default export format** Select a file format. The default export format is text. This format is used only if you do not specify a FORMAT clause in an OUTPUT statement.
- **Default import format** This option is not supported for use with Sybase IQ databases.

  See Chapter 7, “Moving Data In and Out of Databases” in the *System Administration Guide: Volume 1*.
- **Text options** Specify the default symbols used for the field separator, quote string, and escape character when you import or export data in text format.
  - **Default field separator** The symbol used to separate values in text files. The default value is a comma (,).
  - **Default quote string** The symbol used to enclose strings in text files. The default value is a single quote (').
  - **Default escape character** The symbol used in place of unprintable characters in text files. The escape character must be one single-byte character. The default value is a backslash (\).

- **Default encoding** The encoding used when importing and exporting files. If you change this value, it is changed only for the current Interactive SQL session. When you start a new Interactive SQL session, it is restored to its default value. The default value is (Default). If you select (Default), the encoding is determined as follows:
  - The code page specified in the ENCODING clause of the INPUT, OUTPUT, or READ statement.
  - The default code page for the computer Interactive SQL is running on.

Options window: Messages

The Messages page of the Options window in Interactive SQL has these components:

- **Measure execution time for SQL statements** Select this option if you want Interactive SQL to measure the time it takes for a statement to execute. The time appears on the Messages tab. By default, this option is selected.
- **Show separate Messages pane** Select this option if you want information from the database server, such as execution time, to appear in the Messages pane between the SQL Statements and Results panes rather than on the Messages tab in the Results pane. By default, database server information appears on the Messages tab in the Results pane.
- **Default number of lines in Messages pane** Type the number of lines you want returned in the Messages pane. The default number is 7. If you select Show Separate Messages Pane, the number is also the height (in lines) of the Messages pane.

Options window: Source Control

The Source Control page of the Options window in Interactive SQL has these components:

- **Enable source control integration** Select this option if you want Interactive SQL to integrate with the source control system on your computer when you work with files.

    - **Configure** Displays the Custom Source Control Options window where you can view and edit command line actions.

- **Automatically check out files when editor contents are modified** Select this option to automatically check source files out of the source control program when editor contents are modified.

Options window: Toolbar

The Toolbar page of the Options window in Interactive SQL has these components:

- **Execute all statements** Executes all text in the SQL Statements pane. This is the default. You can also perform this operation by pressing F5 or choosing SQL > Execute.
- **Execute selected statements** Executes the text that is selected in the SQL Statements pane. If no text is selected, all the statements are executed. You can also perform this operation by pressing F9 or choosing SQL > Execute Selection.

Options window: Editor

The Editor page of the Options window in Interactive SQL allows you to configure the appearance of text typed in the SQL Statements pane. The page has several tabs.

Editor tab

The Editor tab allows you to configure the appearance of text typed in the SQL Statements pane.

- **Vertical scroll bar** Shows or hides a vertical scroll bar when the window is too small to contain all the text.
- **Horizontal scroll bar** Shows or hides a horizontal scroll bar when the window is too small to contain all the text.

Tabs tab

The Tabs tab has these components:

- **Tab size** Lets you set the tab size (in number of spaces).

- **Indent size** Lets you set the size of indents (in number of spaces).
- **Insert spaces** Inserts *n* spaces when you press the Tab key instead of inserting one tab character. The value for *n* will be between one and the number of indent size spaces, depending on how many spaces are required to move the cursor forward to the next tab stop.
- **Keep tabs** Inserts a tab character into the document and moves the cursor forward to the next tab stop when you press the Tab key.
- **Auto indent** Lets you set the autoindent feature, which includes these options:
  - **None** Disables autoindent.
  - **Default** Uses the tab and indent sizes that are set.
  - **Smart** Uses previous lines of code as a guideline for indenting open and closing braces.

    •**Indent opening brace** Select this option to indent opening braces. This option is enabled when the Smart option is selected.

    •**Indent closing brace** Select this option to indent closing braces. This option is enabled when the Smart option is selected.

Format tab

The Format tab has these components:

- **Text Highlighting** Lets you specify the color and style of different types of text in the main editing window. Choose a type of text and then set the foreground, background, and style for that text type.
- **Font** Lets you specify the font that appears in the SQL Statements pane.
- **Font size** Lets you specify the font point size of the text that appears in the SQL Statements pane.
- **Caret color** Lets you specify the color of the blinking on-screen cursor indicator.
- **Sample** Shows an updated sample of the text with the settings you configure above.
- **Reset All** Returns all settings to their default values.

Print tab

The Print tab has these components:

- **Header** Lets you specify header information, as well as how it is formatted, when you print the contents of the SQL Statements pane. By default, header text is left-aligned. Press the > button for a list of available options.
- **Footer** Lets you specify what information appears in the footer, as well as how it is formatted, when you print the contents of the SQL Statements pane.By default, footer text is left-aligned. Press the > button for a list of available options.
  - **> button** Choose from the following options for the header or footer:
    - File Name
    - File Time
    - File Date
    - Page Number
    - Page Count
    - Current Time
    - Current Date
    - Left Align
    - Center
    - Right Align

    The items that you select need not all use the same alignment. For example, you can choose to left-align the file name and right-align the date in the header. By default, all text in both the header and the footer is left-aligned. You must specify the alignment before the type of text. For example, to center the file name in the header, type &C&F in the Header field, or press the > button and select the Center option and then press the > button again and select the Filename option.

    You can also enter text to appear in the header and footer fields. For example, if you type Page &P of &p in the Footer field, Page 1 of 1 appears in the footer of the printed document.
- **Font size** Lets you select the font point size for the printed text.

Text Completion tab

Use the Text Completion tab to configure settings for the text completion list.

- **What objects do you want in the list?** Select the objects you want to appear in the text completion window. By default, the names of all database objects appear in the text completion list.
    - **Tables and views** Select this option for tables and views to appear in the text completion list.
    - **Columns** Select this option for columns to appear in the text completion list.
    - **Stored procedures** Select this option for stored procedures to appear in the text completion list.
    - **SQL functions (e.g. ABS. COUNT, SUBSTRING)** Select this option for built-in SQL functions to appear in the text completion list.
- **Show system objects** Select this option if you want system objects to appear in the text completion list.
- **Show owner names** This option controls whether an object’s owner is displayed in the list of objects. It does not control whether the owner name appears in the completed text.
- **Show tool tips** Select this option for tooltips to appear in the text completion window.
- **Quote identifiers** Select this option to enclose identifiers in double quotes. Identifiers must be enclosed in double quotes if any of the following conditions are true:
    - The identifier contains spaces.
    - The first character of the identifier is not an alphabetic character.
    - The identifier contains a reserved word.
    - The identifier contains characters other than alphabetic characters and digits.
- **Automatically complete text** Select this option to automatically complete identifier names, as long as there is only one match. If multiple identifiers match the text you type, you can choose the one you want from a list.

Options window: Sybase IQ

Note On SUN platforms, the Sybase IQ page is labelled SQL Anywhere when you access the Options window using Sybase Central. The option is labelled Sybase IQ on SUN platforms when you access the Options window.

Use the Sybase IQ page to configure options for SQL results, options for SQL commands, Query Editor settings, and options for checking for Sybase IQ software updates. The page includes several tabs.

Results tab

The Display area of the Results tab has these components:

- **Display null values as** Specify how you want nulls to appear in table columns. You can use any string for this value. The default value is (NULL). If this field is blank, null values appear as an empty string.
- **Maximum number of rows to display** Specify the maximum number of rows that can appear in the Results pane. The default is 500.
- **Truncation length** Specify the number of characters that can appear in each column in the Results pane. If you specify a value of 0, the columns are not truncated. The default is 256.

The Style area of the Results tab has these components:

- **Scrollable table** Display the result set in a scrollable table. You can edit the result set in this format. This is the default. The following options are enabled when you select Scrollable table:
  - **Automatically refetch results** Select this option to automatically regenerate the result set after you execute an INSERT, UPDATE, or DELETE statement. This is on by default.
  - **Show row number** Select this option to display row numbers beside your result set in the Results pane. By default, this option is selected.
  - **Disable editing** Select this option to make result sets read-only. This option applies to result sets of queries that you execute after selecting this option.
  - **Automatically release database locks** Select this option to release the locks created during your transaction.

When this option is selected, after you execute a statement that returns a result set, Interactive SQL checks if your connection has any uncommitted changes in the database. If none exist, then Interactive SQL releases your schema locks; otherwise, Interactive SQL does not release your schema locks. That is, Interactive SQL does not release your schema locks if you have any uncommitted changes to the database.

- **Font** Select one of the following options to specify the font that is used for table data in the Interactive SQL Results pane:

  •**System font** Select this option to use the computer's normal text font. This is the default.

  •**Editor font** Select this option to use the same font as the SQL Statements pane.

  •**Custom font** Select this option to specify the font, font style, and point size you want to use. Click Browse to select the settings in the Font window

- **Text** Display the result set as text using a monospaced font. The result set is not editable in this format.

The Results Processing area of the Results tab has these components:

- **Show results from the last statement** Prints the result set from the last statement. This is the default.
- **Show results from each statement** Prints the result set for each statement.
- **Show only the first result set** Prints the first result set returned. This is the default.

- **Show all result sets** Prints all result sets. Select this option to display multiple result sets in the Results pane when you execute a procedure that returns multiple SELECT statements. When you are running in windowed mode, each result set appears on a separate tab in the Results pane. The setting of this option also applies when you are running Interactive SQL as a command line program. By default, Interactive SQL does not display multiple result sets.

Note Interactive SQL may run out of memory when displaying large result sets, if you do any of the following:

- Set the Maximum Number of Rows to Display option to All Rows
- Select Show Results From Each Statement
- Select Show All Results Sets

If Interactive SQL runs out of memory, it reports the problem but does not display the result set.

Commands tab

The following options let you select when to commit changes to the database. You can also manually commit by entering an explicit COMMIT statement whenever appropriate.

- **Commit after every statement** Commit changes to the database after each SQL statement is executed.
- **Commit on exit or disconnect** Commit changes to the database when you exit your Interactive SQL session. This is the default.

Query Editor tab

The Query Editor tab has these components:

- **Fully qualify table and column names** Select this option to fully qualify table and column names with their owner names when you are constructing queries in the Query Editor.
- **Quote names** Select this option to enclose the names of identifiers in double quotes when you are constructing queries in the Query Editor.

- **Get list of tables on startup** Select this option to populate tables when you open the Query Editor. This is the default. If you are connected to a database with a large number of tables or have a slow communications link, you may want to turn this option off so the Query Editor opens more quickly.

Check for Updates tab

You can configure whether Sybase IQ should check for software updates and how often it should do so. Checking for updates is done when you open Interactive SQL, Sybase Central, or Sybase IQ.

You can check for updates any time from the Interactive SQL Help menu.

This tab has the following components:

- **When to check for updates** Choose one of the following options to specify how often Sybase IQ should check for updates. The default is Never.
    - **On application startup** Check for updates each time Interactive SQL, Sybase Central, or Sybase IQ is started.
    - **Daily** Check for updates each time Interactive SQL, Sybase Central, or Sybase IQ is started each day.
    - **Weekly** Check for updates each time Interactive SQL, Sybase Central, or Sybase IQ is started each week.
    - **Monthly** Check for updates each time Interactive SQL, Sybase Central, or Sybase IQ is started each month.
- **What to check for** Choose any combination of the following options to specify what types of updates Sybase IQ should check for. By default, the following options are selected:
    - **Express bug fix** Check for express bug fixes.

      An express bug fix is a subset of the software with one or more bug fixes. The bug fixes are listed in the release notes for the update. Bug fix updates can be applied only to installed software with the same version number. Some testing has been performed on the software, but the software has not undergone full testing. Do not distribute these files with your application unless you have verified the suitability of the software yourself.
    - **Maintenance release** Check for maintenance releases of the software.

A maintenance release is a complete set of software that upgrades installed software from an older version with the same major version number (version number format is major.minor.patch.build). Bug fixes and other changes are listed in the release notes for the upgrade.

- **Other information** Check for other information, such as new product releases or upcoming events.

Leaving dbisql

When you have finished working with dbisql, the EXIT command returns you to the operating system.

The Interactive SQL Classic (dbisqlc) utility

Note Sybase recommends that you use the dbisql utility instead of dbisqlc. The dbisqlc utility may not be supported in future releases.

The dbisql command runs Interactive SQL Java. The Java version has many advantages and more features, including multiple windows, each opening a separate database connection. The Interactive SQL Classic utility described in this section is called dbisqlc to distinguish it from the Java version. For information about the Java version, see "The Interactive SQL (dbisql) utility" on page 43.

dbisqlc provides the user with an interactive environment for database browsing and for sending SQL statements to the database server. You can either type SQL commands, or run command files. dbisqlc also provides feedback about the number of rows affected, the time required for each command, the execution plan of queries, and any error messages.

For dbisql to function correctly, $HOME must exist and must be writable by the user.

Syntax

dbisqlc [*options*] [*dbisqlc-command* | *command-file*]

Parameters

Table 2-5 lists the available options for dbisqlc.

Table 2-5: dbisqlc options

Option	Description
-c *"keyword=value; ..."*	Supply database connection parameters.
-d *delimiter*	Specify command delimiter.
-q	Quiet mode—no windows or messages.
-r	Return the error “Not enough fields allocated in sqlda,” if the defined result set of the stored procedure does not match the actual result set.
-x	Check syntax only—no commands executed.

Usage

To customize dbisqlc, change to your home directory ($HOME) and issue the following command to copy the *default.tix* file into it:

```
% cp $IQDIR15/tix/default.tix .
```

This file controls key sequences for dbisqlc and improves the appearance of the command window. See “Shift and Ctrl Keys” on page 84.

dbisqlc-command* | *command-file If *dbisqlc-command* is specified, then dbisqlc executes the command. You can also specify a command file name. If you do not specify a *dbisql-command*, dbisqlc enters the interactive mode, where you can type a command into a command window. In Windows, you can enter interactive mode by selecting Start > Programs > Sybase > Sybase IQ 15.1 > Interactive SQL Classic.

-c "*keyword=value*; ..." Specify connection parameters. See Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*. If you do not specify any connection parameters, the environment variable SQLCONNECT is used. If Interactive SQL cannot connect, enter the appropriate parameters in the dialog box that appears.

Note Sybase recommends that, to avoid ambiguity, you always specify connection parameters for dbisqlc instead of relying on defaults, whether you specify them in a command line or an initialization file such as *.odbc.ini* on UNIX, or *odbc.ini* on Windows. If you start more than one database on a server, for example, specify the database name, and in a network with subnets, specify the communications protocol parameter with host number. See Chapter 3, “Sybase IQ Connections” in the *System Administration Guide: Volume 1*.

-d Specify a command delimiter. By default, the delimiter is the semicolon.

-q Do not display output messages. This option is useful only if you start Interactive SQL with a command or command file.

-r Returns the error "`Not enough fields allocated in sqlda`" if the defined result set of the stored procedure does not match the actual result set. This option may be useful when you are querying stored procedures.

-x Scan commands but do not execute them. You may find this option useful for checking long command files for syntax errors.

Examples

The examples in this section illustrate how to use dbisqlc from the system command prompt. The rest of the dbisqlc sections in this chapter discuss how to use dbisqlc interactively.

- The following command, entered at a system prompt, runs the command file *mycom.sql* against the current default server, using the user ID dba and the password sql. Note that the default password is case-sensitive but the user ID is not.

```
dbisqlc -c "uid=DBA;pwd=sql" mycom.sql
```

- The following command, entered at a system prompt, adds a user to the current default database:

```
dbisqlc -c "uid=DBA;pwd=sql" grant connect to joe
identified by passwd
```

See also

For detailed descriptions of SQL statements and dbisql commands, see Chapter 1, "SQL Statements" in*Reference: Statements and Options*.

Starting dbisqlc

Once a database server is running, you can connect to the server from dbisqlc. This example describes how to connect to the demo database iqdemo.

❖ Starting dbisqlc and connecting to a database from the command prompt

1 Ensure that the sample database is loaded on a running server, as described in "Starting the database server" in Chapter 3, "Running and Connecting to Servers" in the *Introduction to Sybase IQ*.

2 At the command prompt on your UNIX or Windows system, enter:

```
dbisqlc -c "uid=dba;pwd=sql"
```

The -c parameter specifies connection parameters. See Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

Note The default user ID and password for the iqdemo database and other IQ databases are “dba” and “sql.” If you have trouble connecting to a database, try changing the case of the user ID and password that you supply.

❖ Starting dbisqlc and connecting to a database on Windows

1 Start the server, as described in “Starting the database server” in Chapter 3, “Running and Connecting to Servers” in the *Introduction to Sybase IQ*.

2 Select Start > Programs > Sybase > Sybase IQ 15.1 > Interactive SQL Classic.

The dbisqlc Connect window appears.

3 On the Login tab, enter the user ID:

```
dba
```

and the password:

```
sql
```

This is the default user ID and password for Sybase IQ databases when they are created. (The password does not appear when you type it.) By default, the password is case-sensitive for a case-sensitive database; the user ID is not.

4 Click the Database tab and type the server name (for example, “*hostname_iqdemo*” for the demo database). If your server is on a remote machine, make sure that you select “Search network for database servers.”

5 On the Network tab, select TCPIP.

If your database is on a remote machine, you must add host information in the space beside TCPIP by typing “host=*servername*:*nnnn*” where *servername* is the name of your system and *nnnn* is your port number. (The default port number is 2638, but if the host was started with a different number, use that instead.)

6 Click OK to connect to the database.

The dbisqlc command window

After you enter the dbisqlc command and connect to a database, the dbisqlc command window appears. The appearance and functionality of this window differ on UNIX and Windows systems.

dbisqlc Command window on UNIX

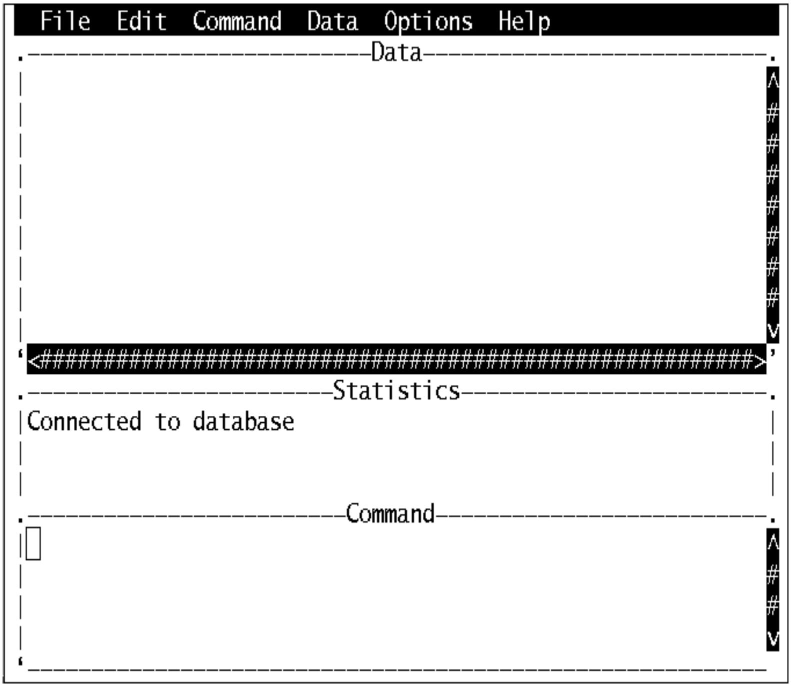
If you run Interactive SQL Classic on Windows, skip to “dbisqlc Command window on Windows” on page 86.

In UNIX, the dbisqlc window is divided into three sections:

- Data section, at the top
- Statistics section, in the middle
- Command section, at the bottom

If you follow the instructions in “Starting dbisqlc” on page 80 correctly, you see the message “Connected to database” in the Statistics section.

Figure 2-2: dbisqlc window on UNIX



```
Connected to database
```

The dbisqlc command window on UNIX varies according to the operating system. For example, on Solaris systems, the Common Desktop Environment window differs from that on OpenWindows. The dbisqlc command window is divided into three subwindows. The top section is entitled Data, the middle Statistics, and the bottom section is Command, as shown in Figure 2-2.

Instructions in this chapter may tell you to choose items from pull-down menus. These menus are located at the top of the screen.

Executing commands

Type commands in the Command section, and execute them by pressing the F9 function key.

Note If your terminal does not support the function keys, type Ctrl+F followed by the function key number. For example, for the F9 function key, type Ctrl+F+9. If you still have trouble executing commands, set up the terminfo extension file as described in “Shift and Ctrl Keys.”

Enter multiple commands by separating them with semicolons.

You can store commands to an ASCII file by typing Ctrl+A+F to activate the File menu, and using the down arrow to choose File > Save from the menu bar.

To load from an ASCII file, type Ctrl+A+F and choose File > Open from the menu bar.

Function keys

dbisqlc uses some function keys and special keys on UNIX as described in Table 2-6.

Table 2-6: dbisqlc function and special keys on UNIX

Function key	Description
F5	Move data to the left by one column in the data window.
Shift+F5	Move data to the left by one character.
F6	Move data to the right by one column.
Shift+F6	Move data to the right by one character.
F7	Display a list of the tables in the database. Use the up and down arrow keys to scroll through the table names changing the highlighted table name. While the list appears, press Enter to insert the current table name into the command window at the cursor position. While the list appears, press F7 to see a list of columns for the highlighted table. Again, use Enter to select the highlighted column name and put it into the command window at the cursor position.
Ctrl+PgUp	Move to the top of data.
Ctrl+PgDn	Move to bottom of data.

Shift and Ctrl Keys

You may need to define key sequences for Shift and Ctrl to be applied to the next key. This enables you, for example, to use Shift+<function key> or Ctrl+PgDn key sequences.

Some terminals do not allow dbisqlc to distinguish when the ALT key is being used. To enter an ALT- key in dbisqlc, use Ctrl+A followed by the key to which the ALT modifier is to be applied. For example, to see the File menu, type Ctrl+A+F. If you use dbisqlc from a remote terminal, such as a VT100 emulator, you may be able to configure the terminal to use “emacs mode” so that sends ALT-<key> as ESC <key>. dbisqlc will also recognize these escape sequences. You can type ESC <key> yourself, but they must be typed in quick succession or the keys will be interpreted individually.

These sequences are controlled by a terminfo extension (*.tix*) file. dbisqlc first looks for *${TERM}.tix* in *${HOME}*, */opt/IQ-15_1/tix*, */opt/IQ-15_1/bin*, and then throughout *${PATH}*. If *${TERM}.tix* is not found, dbisqlc searches for *default.tix* in the same directories. A sample *default.tix* file comes with Sybase IQ.

Displaying data in dbisqlc

You can use dbisqlc is to look at information in databases.

The instructions and examples in this section use a database for a fictional company. The sample database contains information about employees, departments, sales orders, and so on.

All this information is organized into a number of tables that are made up of rows and columns.

Listing the tables in a database

Sybase IQ provides a number of built-in functions called system stored procedures. Use these procedures to display information about your database.

For example, use the sp_iqtable stored procedure to list the tables in the sample database. To run a stored procedure, type its name (and any parameters) in the command window.

See Chapter 7, “System Procedures” in *Reference: Building Blocks, Tables, and Procedures*.

Listing all the columns and rows of the employee table

To list the data stored within database tables, type a SELECT command in the dbisqlc command window, then press F9 (if supported), or Command > Execute from the menu to carry out the command. The example output that follows shows the first several columns and rows of the results of the query which appear in the dbisqlc data window.

Enter:

```
SELECT *
FROM Employees
```

Employee ID	ManagerID	Surname	GivenName	...
102	501	Whitney	Fran	...
105	501	Cobb	Matthew	...
129	902	Chin	Philip	...
148	1293	Jordan	Julie	...
160	501	Breault	Robert	...
...				

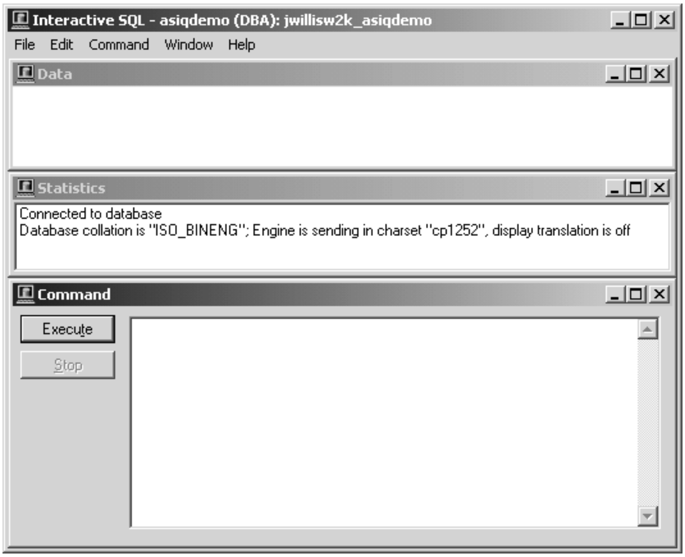
Notes

- In the examples, SQL keywords are shown in uppercase; however, SQL statements are case-insensitive.
- In many examples, each clause of a SQL statement is placed on a separate line for readability. You can enter SQL statements either all on one line, or you can break them over several lines.

dbisqlc Command window on Windows

After you start dbisqlc, the Interactive SQL Classic window opens. It is divided into three subwindows. The topmost is entitled Data, the second Statistics, and, at the bottom of your screen is Command, as shown in Figure 2-3.

Figure 2-3: dbisqlc window on Windows



Connected to database
Database collation is "ISO_BINENG"; Engine is sending in charset "cp1252", display translation is off

Entering commands

If you type more lines than will fit in this window, fit in the window, you can automatically scroll the window, either using the cursor keys or the scroll bar on the right side of the window. You can also size the window, and minimize or maximize it using the standard Windows controls.

Execute commands by pressing the execute key (F9) or clicking the Execute button.

You can store commands to an ASCII file by choosing File > Save. To load them from an ASCII file, choose Open from the menu bar.

Displaying data in dbisqlc

You can use dbisqlc is to look at information in databases.

The instructions and examples in this section use a database for a fictional company. The sample database contains information about employees, departments, sales orders, and so on.

All this information is organized into a number of tables that are made up of rows and columns.

Listing all the columns and rows in the employee table

You display information from a database using the SELECT statement. The following example shows the command to type in the dbisqlc command window. Once you have typed the command, click Execute to carry out the command. The example displays the first several columns and rows of the results of the query, which appear in the dbisqlc data window.

Enter:

```
SELECT *
FROM Employees
```

Employee ID	ManagerID	Surname	GivenName	...
102	501	Whitney	Fran	...
105	501	Cobb	Matthew	...
129	902	Chin	Philip	...
148	1293	Jordan	Julie	...
160	501	Breault	Robert	...
...				

Notes

- In the examples, SQL keywords are shown in uppercase; however, SQL statements are case-insensitive.
- In many examples, each clause of a SQL statement is placed on a separate line for readability. You can enter SQL statements either all on one line, or you can break them over several lines.

The dbisqlc Data window displays a set of rows and columns containing information about the employees. Each row contains information about one employee, and each column contains a particular piece of information for all employees.

Scrolling the data window

The visible portion of the dbisqlc data window cannot hold the entire Employees table. It does not display all the information about each employee, and does not display the entire list of employees.

To see information about each employee (other columns) use the scroll bar at the bottom of the data window. This is a standard Windows scroll bar.

To see more information on other employees (other rows), use the scroll bar to the right of the data window. The employee table in the sample database contains approximately 75 employees.

Sometimes, the vertical scroll bar behaves slightly differently than standard scroll bars, as the number of rows in the result may be unknown. In this case, a guess as to the number of rows is used. If dbisqlc determines that its guess is wrong, the guess is adjusted and the slider "jumps."

Using command recall in dbisqlc

1 Enter:

```
SELECT * FROM Departments
```

2 Press F9.

The contents of the department database table appear in the Data window. As you execute commands with dbisqlc, they are saved in a command history.

3 To recall commands, choose Command > Recall from the menu bar to activate the command recall window.

4 The command recall window displays the first line of the last 15 commands executed.

5 Position the cursor on the first command that you executed, which was:

```
SELECT *
FROM Employees
```

and press Enter. The cursor returns to the command window with the selected command in it. You can either reexecute that command, or modify it to make a new command.

Alternatively, there are several key sequences, listed in Table 2-7, you can use to recall previous commands.

Table 2-7: dbisqlc recall keys

Key sequence	Description
Ctrl+r	Brings up the command recall window
Ctrl+p	Cycles backwards through previously executed commands. Retrieved commands are placed into the command window
Ctrl+n	Cycles forward through previously executed commands

Function keys

Table 2-8 lists function keys and special keys that are available on Windows.

Table 2-8: dbisqlc function and special keys on Windows

Function key	Description
F5	Move data to the left by one column in the data window.
Shift+F5	Move data to the left by one character.
F6	Move data to the right by one column.
Shift+F6	Move data to the right by one character.
F7	Display a list of the tables in the database. Use the up and down arrow keys to scroll through the table names changing the highlighted table name. While the list appears, press Enter to insert the current table name into the command window at the cursor position. While the list appears, press F7 to see a list of columns for the highlighted table. Again, use Enter to select the highlighted column name and put it into the command window at the cursor position.
F9	Execute the command that is in the command window. This operation can also be performed with the mouse by clicking Execute.
F10	Activate the menus at the top of the window.
Page Up	Move data up a page.
Page Down	Move data down a page.
Ctrl+PageUp	Move to top of data.
Ctrl+PageDown	Move to bottom of data.

Canceling a dbisqlc command on Windows and UNIX

Use the Stop button to cancel a command. On UNIX, click CTRL+C to cancel a command.

Stop cancels current processing and prompts for the next command. If a command file is being processed when you click Stop, then select one of: Stop, Continue, or Exit.

Reported errors

When you cancel a command, one of three different errors is reported, depending upon when the end is detected.

1 If the cancellation is detected when dbisqlc is processing the request (as opposed to the database engine), you see:

```
dbisql command terminated by user
```

dbisqlc stops processing immediately and the database transaction is left alone.

2 If the cancellation is detected while the database engine is processing a data definition command (CREATE, DROP, ALTER, and so on,) you see:

```
Terminated by user -- transaction rolled back
```

Since data definition commands all perform a COMMIT automatically before the command starts, the effect of the ROLLBACK is to just cancel the current command.

This message also occurs when the database engine is running in bulk operations mode executing a command that modifies the database (INSERT and DELETE). In this case, ROLLBACK cancels not only the current command, but everything that has been done since the last COMMIT or ROLLBACK. In some cases, it may take a considerable amount of time for the database engine to perform the automatic ROLLBACK.

3 If the cancel is detected by the database engine while processing a standard data manipulation command (SELECT, INSERT, DELETE, and so on,) and the engine is not running in bulk operations mode, you see.

```
Statement interrupted by user.
```

The effects of the current command are undone, but the rest of the transaction is left intact.

CHAPTER 3 Database Administration Utilities

About this chapter

Sybase IQ includes a set of command-line utility programs for performing database administration tasks. This chapter provides reference information for most of the database administration utilities. Utility programs that start and stop the database server are described in Chapter 1, “Running the Database Server”.

Contents

Administration utilities overview

This chapter contains reference information on the programs and database administration utilities that are part of Sybase IQ. The utilities can be accessed from Interactive SQL, or as command-line programs.

For an introduction to Interactive SQL, also called dbisql, see Chapter 2, “Using Interactive SQL (dbisql)”.

For comprehensive documentation about Sybase Central, see the Sybase Central online help. For an introduction to the Sybase Central database administration tool, see Chapter 3, “Running and Connecting to Servers” in the *Introduction to Sybase IQ*.

The administration utilities use a set of system environment variables which are described in Chapter 1, “File Locations and Installation Settings” in *Reference: Building Blocks, Tables, and Procedures*.

Alternatively, you can use a set of SQL statements that perform some administration tasks.. These statements are documented in Chapter 1, “SQL Statements” in *Reference: Statements and Options*.

Backup utility (dbbackup)

The Backup utility makes a copy of the transaction log of a running IQ database, and lets you truncate the transaction log, freeing disk space and improving recovery speed, without having to stop and restart your server.

Note To back up an entire Sybase IQ database, always use BACKUP instead of dbbackup. BACKUP backs up all database files, and is the only way to back up the catalog store. See BACKUP statement in Chapter 1, “SQL Statements” in *Reference: Statements and Options*.

Syntax

dbbackup [*options*] *target-directory*

Parameters

Table 3-1 lists the available options for the dbbackup utility.

Table 3-1: dbbackup options

Option	Description
@*data*	Read options from the specified environment variable or configuration file
-c "*keyword=value*; ... "	Supply database connection parameters
-l (lowercase L) *file*	Send a live backup of the transaction log to a file
-q	Quiet mode — do not print messages
-r	Copy the old transaction log to a new name and start a new empty log
-xo *filename*	Truncate (delete and restart) the transaction log

Usage

The Backup utility allows you to back up the transaction log while other applications or users are using the database. Backup file names are the same as the database file names.

The dbbackup utility truncates the database name to 70 characters and creates a target file with a truncated name. Sybase IQ uses dbbackup when synchronizing secondary servers. Due to dbbackup restrictions, database names must be less than 70 characters long.

If you have adequate disk space, use -r to preserve the existing log file under a new name and start a new empty log. If disk space is limited, use -xo instead to truncate the existing log.

Exit codes are 0 (success) or nonzero (failure).

Options

@data Reads in options from the specified environment variable or configuration file. If both exist, and share the same name, Sybase IQ uses the environment variable.

For more information about configuration files, see the *Installation and Configuration Guide*.

To protect passwords or other information in the configuration file, use the File Hiding utility (dbfhide) to obfuscate configuration file contents. See “File Hiding utility (dbfhide).”

Connection parameters (-c) If you do not specify the connection parameters, connection parameters from the SQL CONNECT environment variable are used, if set. The user ID must have DBA authority or REMOTE DBA authority.

See Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

Live backup (-l (lowercase L)) Enables a secondary system to be brought up rapidly in the event of server failure. A live backup does not terminate, but continues while the server runs. It runs until the primary server becomes unavailable. At that point, it shuts down, but the backed up log file is intact and can be used to quickly bring up a secondary system.

The live backup of the transaction log is always the same length or shorter than the active transaction log. When a live backup is running and another backup restarts the transaction log (dbbackup -x), the live backup automatically truncates the live backup log and restarts the live backup at the beginning of the new transaction log.

See "Live backup of transaction log" in Chapter 12, "Data Backup, Recovery, and Archiving" in the *System Administration Guide: Volume 2*.

Log output messages to file (-o) Write output messages to the named file.

Operate quietly (-q) Do not display output messages. This option is available only when you run this utility from a command prompt.

Rename and start new transaction log (-r) Forces a checkpoint and these steps to occur:

- Copies and saves the current working transaction log to the directory specified in the command.
- Keeps the current transaction log in its current directory, but renames it using the format *yymmddxx.log*, where *xx* are sequential characters starting at *AA* and running to *ZZ*, and *yymmdd* represents the current year, month, and day. This file is then no longer the current transaction log.
- Generates a new transaction log file that contains no transactions. The new file has the name of the former current transaction log and becomes the current transaction log.

Back up the transaction log file only (-t) This can be used as an incremental backup since the transaction log can be applied to the most recently backed up copy of the database files.

Delete and restart the transaction log without a backup (-xo) Delete the current transaction log and start a new one. This operation does not carry out a backup; its purpose is to free up disk space.

target-directory The directory to which the backup files are copied. If the directory does not exist, Sybase IQ creates it. The parent directory must exist.

Examples

The following Windows command backs up the transaction log from the iqdemo database running on the *sample_server* server into the directory *iqbackup*, connecting as user ID DBA with password sql:

```
dbbackup -c
"eng=sample_server;dbn=iqdemo;uid=DBA;pwd=sql"
c:\sample\iqbackup
```

The following example truncates the log of a live server, renames the old log prior to truncation, and creates a exact copy of the log in the target directory:

```
dbbackup -t -r -o dbbackup.out -c
"eng=taj_iqdemo;dbn=iqdemo;uid=DBA;pwd=sql"
logbackup
```

```
dbbackup -xo -c
"eng=sample_server;dbn=iqdemo;uid=DBA;pwd=sql"
```

CP874toUTF8 utility

The CP874toUTF8 utility converts data in the CP874 character set into a UTF8 collation that is supported by Sybase IQ for the Thai language. You can also use this utility to load data in the CP874 character set without converting it to UTF8.

Syntax

CP874toUTF8 *[CP874InputFile]*

Usage

You can run this utility only from the command prompt.

CP874toUTF8 reads the named file in the CP874 character set (or standard input if no files are named) and prints the UTF8 conversion to standard output.

Note Files with embedded NULL characters (‘\0’) are not converted correctly. Remove such characters before running this utility.

Use caution with large data files; the UTF8 output can be up to three times larger than the input data. Input and output file size must both be within operating system limits.

CP874toUTF8 returns a 0 exit status upon successful completion. A nonzero exit status indicates an error occurred. The CP874toUTF8 utility writes all error messages to stderr.

CP874toUTF8 calls the International Components for Unicode (ICU) library to perform the data conversion. If ICU cannot convert the data, CP874toUTF8 reports the conversion error "`Conversion failed from CP874 to UTF8.`" If the conversion fails, divide the file into smaller sections for conversion, to isolate the point of failure. If you cannot resolve the problem, contact Sybase Technical Support.

Data Source utility (iqdsn)

The Data Source utility is a cross-platform alternative to the ODBC Administrator for creating, changing, deleting, describing, and listing Sybase IQ ODBC data sources. On Windows operating systems, data sources are held in the Registry. On UNIX operating systems, data sources are held in the *.odbc.ini* file. Use -iqdsn for batch operations.

Syntax

iqdsn [*modifier-options*]
{ **-l**
| **-d** *dsn*
| **-g** *dsn*
| **-w** *dsn* [*details-options;...*]
| **-cl** }

Parameters

The following tables list the available options for the iqdsn utility.

Table 3-2: iqdsn major options

Major option	Description
-l	List either all data sources.
-d *dsn*	Delete the named data source.
-g *dsn*	List (get) details about the named data source.
-w *dsn* [*details-options*]	Create (write) a data source definition.
-cl	List available connection parameters.

Table 3-3: iqdsn modifier-options

Modifier option	Description
-b	Brief. Print connection string for the data source.
-q	Quiet. Do not print banner.
-v	Verbose. Print connection parameters in tabular form.
-va	Verbose All. Print connection parameters in same format as -v, but also include other hidden parameters. Use this option to display ODBC driver qualifier needed for remote data access on those UNIX platforms that support such access, or for some third-party driver managers.

Modifier option	Description
-y	Delete or overwrite data source without confirmation.

Table 3-4: iqdsn details-options

Details option	Description
-c "keyword=*value*;..."	Supply database connection parameters
-ec *encryption type*	Encrypt all network packets
-o *filename*	Write client message to file name
-p *size*	Set maximum network packet size
-r	Disable multiple record fetching
-tl *seconds*	Client liveness timeout period
-x *list*	List network drivers to run
-z	Display debugging information
server-name	Connect to named database server

Usage

The iqdsn modifier options can occur before or after the major option specification. The order makes a difference only if you specify a a connection parameter value more than once. In such a case, the last value specified is used.

Major options

List defined data sources (-l) Lists the available Sybase IQ ODBC data sources. You can modify the list format using the -b or -v options.

Delete the named data source (-d) Deletes the named data source. If you supply -y, any existing data source is overwritten without confirmation.

List (get) details of the named data source (-g) List the definition of the named data source. You can modify the format of the output using the `-b` or `-v` option.

Create (write) a data source definition (-w) Creates a new data source, or overwrites one if one of the same name exists. If you supply `-y`, any existing data source is overwritten without confirmation.

List available connection parameters (-cl) This convenience option lists the connection parameters supported by the iqdsn utility.

Modifier options

Print connection string for the data source (-b) Format the output of the list as a single line connection string.

Do not print banner (-q) Suppress the informational banner.

Do not print banner or titles (-qq) Suppress both the informational banner and titles. This option can only be used with the -l and the -cl options.

Print connection parameters in tabular form (-v) Format the output of the list over several lines, as a table.

Delete or overwrite data source without confirmation (-y) Automatically delete or overwrite each file without prompting you for confirmation.

Details options

Connection parameters (-c) Specify connection parameters as a connection string.

See Chapter 4, “Connection and Communication Parameters” in *System Administration Guide: Volume 1*.

Encrypt network packets (-ec) Encrypt packets sent between the client application and the server.

See “Encryption connection parameter [ENC]” in Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

Log output messages to file (-o) Write output messages to the named file. By default, messages are written to the console.

See “LogFile connection parameter [LOG]” in Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

Operate quietly (-q) Do not display output messages. This option is available only from the command-line utility.

Set maximum network packet size (-p) The maximum packet size for network communications, in bytes. The value must be greater than 300, and less than 16000. The default setting is 1492.

See “CommBufferSize connection parameter [CBSize]” in Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

Disable multiple-record fetching (-r) By default, when the database server gets a simple fetch request, the application asks for extra rows. You can disable this behavior by using this option.

See “DisableMultiRowFetch connection parameter [DMRF]” in Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

Set client liveness timeout (-tl) Terminates connections when they are no longer intact. The value is in seconds.

The default is the server setting, which in turn has a default of 120 seconds.

See “LivenessTimeout connection parameter [LTO]” in Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*.

Display debugging information (-z) Provide diagnostic information on communications links on startup.

Server name Connect to the named server. Only the first 40 characters are used.

See Chapter 1, "Running the Database Server."

Examples

Write a definition of the data source newdsn. Do not prompt for confirmation if the data source already exists.

```
iqdsn -y tcpip -w newdsn -c "uid=DBA;pwd=sql" -v
```

You can also change the order of options:

```
iqdsn -w newdsn -c "uid=DBA;pwd=sql" -x tcpip -y
```

List all known user data sources, one data source name per line:

```
iqdsn -l
```

List all data sources along with their associated connection string:

```
iqdsn -l -b
```

Report the connection string for user data source MyDSN:

```
iqdsn -g MyDSN
```

Delete the data source BadDSN, but first list the connection parameters for BadDSN and prompt for confirmation:

```
iqdsn -d BadDSN -v
```

Delete the data source BadDSN without prompting for confirmation.

```
iqdsn -d BadDSN -y
```

Create a data source named NewDSN for the database server MyServer:

```
iqdsn -w NewDSN -c "uid=DBA;pwd=sql;eng=bar"
```

If a NewDSN already exists, the utility asks you if you want to overwrite it.

The following example connects to the sample database server. The server name sample overrides the previous specified value of MyServer:

```
iqdsn -w NewDSN -c "uid=DBA;pwd=sql;eng=MyServer"
sample
```

List all connection parameter names and their aliases:

```
iqdsn -cl
```

See also

"Working with ODBC data sources" and "Using ODBC data sources on UNIX" in Chapter 3, "Sybase IQ Connections" in the *System Administration Guide: Volume 1*.

DBSpace Header utility (iqheader)

Use the DBSpace Header utility to determine which server, if any, is using a particular device, file or LUN (Logical Unit Number) as a dbspace in order to analyze disk usage or to configure a multiplex query server.

The iqheader utility reports the configuration of an arbitrary device regardless of whether it is currently in use by an IQ server.

The user interface is a standalone console application called iqheader (header.exe on Windows). The iqheader tool searches the device for a IQ dbspace header and reports the header information in a user-readable format.

Note LUN is a logical unit number and is used to identify SCSI devices so the host can address and access the data on each disk drive in an array.

Syntax

iqheader [[*dbspace_path*]

Parameters

The iqheader application takes a single parameter, which is the device to be checked.

Usage

iqheader usage When invoked with no parameters, a usage summary is reported and a nonzero status is returned:

```
>iqheader
Usage: iqheader [dbspace_path]
```

iqheader error When the specified target is not an IQ dbspace, an error message is reported and a nonzero status is returned:

```
>iqheader /dev/null
Not an IQ file: Error 0
```

Operating system errors When the specified target is unreadable, or any file operation fails due to an error returned from the operating system, the native operating system error appears and a nonzero status is returned.

```
>iqheader /dev/rdsk/c1t32d0s1              <
Open Failed: No such file or directory

>iqheader /dev/rdsk/c1t3d0s1               <
Open Failed: Permission denied
```

iqheader output When a valid IQ dbspace is specified, iqheader prints the dbspace configuration to the console and returns a 0 exit status. Table 3-5 describes the fields in the iqheader output.

Table 3-5: iqheader output

Field	Description
File Name	Name of the file.
Full Path	Full path after symbolic link resolution
Version	Dbspace file format version
File ID	Unique number assigned to each dbspace
Create Time	Time of dbspace creation
RW Mode	Current read-write mode: RW, RO, RW, N/A (Upgraded)
Last RW Mode	Last dbspace mode
Size (MB)	DBSpace size, in megabytes
Reserve (MB)	DBSpace reserve size, in megabytes
Block Size	Size of IQ block, in bytes
Page Size	Size of IQ page, in bytes
First Block	First IQ block number mapped to dbspace
Block Count	Number of IQ blocks that map to actual disk blocks
Reserve Blocks	Number of IQ blocks that may be added to this dbspace
Last Real Block	Last IQ block number that maps to an actual disk block
Last Mapped Block	Last IQ block number mapped to dbspace
OFlag	Online status (YES/NO)
Create ID	Commit ID in which dbspace was created
Alter ID	Last commit ID in which dbspace was altered
DBID1	Location of first database identity
DBID2	Location of second database identity
DBSpace ID	Unique identifier assigned to each dbspace
_NextFLAllocLowerBank	Lower bound of pre-allocate space for dbspace
_NextFLAllocUpperBank	Upper bound of pre-allocate space for dbspace
Pre-alter commit ID	Commit_id in the system tables ISYSDBFILE and ISYSIQDBSPACE
_ReqNumFreeListBlocks	Number of blocks of type 'F' (free list blocks)

Example

The following example shows output for iqheader:

```
File Name: file1.iq
Full Path: /dev/dsk/file1.iq
DBFile Header Info
Version: 2
File ID: 16395
```

```
Create Time: 2008-06-02 21:57:00
RW Mode: RO
Last RW Mode: RW
Size (MB): 20
Reserve (MB): 20
Block Size: 8192
Page Size: 131072
First Block: 9408960
Block Count: 2560
Reserve Blocks: 2560
Last Real Block: 9411519
Last Mapped Block: 10454399
OFlags: 1
Create ID: 6905
Alter ID: 6964
DBID1: 0
DBID2: 0
DBSpace ID: 16395
_NextFLAllocLowerBank: 0
_NextFLAllocUpperBank: 0
Pre-alter commit ID: 6925
Dropped: NO
```

See also

Chapter 5, "Working with Database Objects," in the *System Administration Guide: Volume 1.*

File Hiding utility (dbfhide)

The File Hiding utility, dbfhide, allows you to add simple encryption to configuration files and initialization files to hide the contents of each file.

Syntax

dbfhide *original-configuration-file encrypted-configuration-file*

Parameters

The following table lists the available options for the dbfhide utility.

Option	Description
original-configuration-file	Name of the original file.
encrypted-configuration-file	Name for the new obfuscated file.

Usage

Configuration files are used by some utilities to hold command-line options. These options can contain a password. You can use the dbfhide utility to add simple encryption to configuration files and *.ini* files used by Sybase IQ and its utilities, and thereby obfuscate the contents of the file. The original file is not modified. Once you add simple encryption to a file, there is no way to remove it. To make changes to an obfuscated file, keep a copy of the original file that you can modify and obfuscate again.

❖ Hiding the contents of .ini files

1 Save the file with a different name.

```
rename asaldap.ini asaldap.ini.org
```

If you do not keep a copy of the original file, then you cannot modify the contents of the file once it has been obfuscated.

2 Obfuscate the file with the File Hiding utility, giving the obfuscated file the required file name:

```
dbfhide asaldap.ini.org asaldap.ini
```

3 Protect the *asaldap.ini.org* file using file system or operating system protection, or store the file in a secure location.

To make a change to the *asaldap.ini* file, edit the *asaldap.ini.org* file and repeat step 2.

Warning! You should not add simple encryption to the *.odbc.ini* system information file with the File Hiding utility (dbfhide) on UNIX unless you will only be using Sybase IQ data sources. If you plan to use other data sources, then obfuscating the contents of the *.odbc.ini* file may prevent other drivers from functioning properly.

dbfhide does not accept the *@data* parameter to read in options from a configuration file.

Examples

Create a configuration file that starts the personal database server and the sample database. The file should set a cache of 10MB, and name this instance of the personal server “Elora”. The configuration file would be written as follows:

```
# Configuration file for server Elora -n Elora -c 10M
path\asademo.db
```

Lines beginning with # are treated as comments.

Name the file *sample.txt*. To start the database using this configuration file, enter:

```
start_iq @sample.txt
```

Add simple encryption to the configuration.

```
dbfhide sample.txt encrypted_sample.txt
```

Use the encrypted_sample.txt file to start a database:

```
start_iq @encrypted_sample.txt
```

For more information about using configuration files, see the *Installation and Configuration Guide*.

The following command adds simple encryption to the *asaldap.ini* file:

```
dbfhide asaldap.ini encrypted_asaldap.ini
```

Information utility (dbinfo)

The command-line Information utility dbinfo displays information about a database catalog store. The information returned by dbinfo does not reflect the IQ store.

Any valid user ID can run dbinfo, but to obtain page usage statistics you need DBA authority.

Syntax

dbinfo [*options*]

Parameters

Table 3-6 lists the available options for the dbinfo utility.

Table 3-6: dbinfo options

Option	Description
-c *"keyword=value; ..."*	Database connection parameters
-o *filename*	Log output messages to a file
-q	Operate quietly
-u	Output page usage statistics

Usage

The dbinfo utility indicates when the database was created, the name of any transaction log file or log mirror that is maintained, the catalog store page size, the version of installed Java classes, and other information. Optionally, it can also provide catalog table usage statistics and details.

Options

Connection parameters (-c) Specify connection parameters. See Chapter 4, "Connection and Communication Parameters" in the *System Administration Guide: Volume 1*.

Log output messages to file (-o) Write output messages to the named file.

Operate quietly (-q) Do not display output messages.

Page usage statistics (-u) Display information about the usage and size of all catalog store tables, including system and user-defined tables.

You can only request page usage statistics if no other users are connected to the database.

Log Translation utility (dbtran)

Use the Log Translation utility, dbtran, to translate a transaction log into a *.sql* command file.

You access the Log Translation utility at the command prompt, using the dbtran command. .

Syntax

Running against a database server:

dbtran [*options*] -c { *connection-string*} -n *SQL-file*

Running against a transaction log:

dbtran [*options*] [*transaction-log*] [*SQL-file*]

Parameters

Option	Description
@data	Reads in options from the specified environment variable or configuration file.
-a	Controls whether uncommitted transactions appear in the transaction log. The transaction log contains changes made before the most recent COMMIT by any transaction. Changes made after the most recent COMMIT are not present in the transaction log. If you do not specify -a, only committed transactions appear in the output file. If you specify -a, any uncommitted transactions found in the transaction log are output followed by a ROLLBACK statement.

Option	Description
-c *"keyword=value; ..."*	Specifies the connection string when running the utility against a database server. See “Connection parameters” in Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*
-d	Specifies that transactions are written in order from earliest to latest. This feature is provided primarily for use when auditing database activity: do not apply dbtran output against a database.

-ek *key*	Specifies the encryption key for strongly encrypted databases. If you have a strongly encrypted database, you must provide the encryption key to use the database or transaction log. Specify either -ek or -ep, but not both. The command fails if you do not specify the correct encryption key. If you are running dbtran against a database server using the -c option, specify the key using a connection parameter instead of using the -ek option. For example, the following command gets the transaction log information about database enc.db from the database server sample, and saves its output in *log.sql*.
	dbtran -n log.sql -c "ENG=sample;DBF=enc.db;
	UID=DBA;PWD=sql;DBKEY=mykey"

-ep	Prompts for the encryption key. This option causes a window to appear, in which you enter the encryption key. It provides an extra measure of security by never allowing the encryption key to be seen in clear text. Specify either -ek or -ep, but not both. The command fails if you do not specify the correct encryption key. If you are running dbtran against a database server using the -c option, specify the key using a connection parameter, instead of using the -ep option. For example, the following command gets the transaction log information about database enc.db from the database server sample, and saves its output in *log.sql*.
	dbtran -n log.sql -c "ENG=sample;DBF=enc.db;
	UID=DBA;PWD=sql;DBKEY=mykey"

-f	Outputs only transactions that were completed since the last checkpoint.

Option	Description
-g	Adds auditing information to the transaction log if the auditing database option is turned on. See “AUDITING option [database]” in Chapter 2, “Database Options” in *Reference: Statements and Options*.
-ir *offset1,offset2*	Outputs a portion of the transaction log between two specified offsets.
-is *source,...*	Outputs operations on rows that have been modified by operations from one or more of the following sources, specified as a comma-separated list: • **All** All rows. This is the default setting. • **SQLRemot**e Include only rows that were modified using SQL Remote. You can also use the short form “SR”. • **RepServer** Include only rows that were modified using the Replication Agent (LTM) and Replication Server. You can also use the short form “RS”. • **Local** Include only rows that are not replicated.
-it *owner.table,...*	Outputs those operations on the specified, comma-separated list of tables. Specify each table as owner.table.
-j date/time	Translates only transactions from the most recent checkpoint prior to the given date or time. The user-provided argument can be a date, time, or date and time, enclosed in quotes. you omit a time, the default is the beginning of the day. If you omit a date, the current day is the default. The acceptable format for the date and time is: "YYYY/MMM/DD HH:NN".
-k	Prevents partial *.sql* files from being erased if an error is detected. If an error is detected while dbtran is running, the *.sql* file generated until that point is normally erased to ensure that a partial file is not used by accident. Specifying this option may be useful if you are attempting to salvage transactions from a damaged transaction log.
-m	Specifies a directory that contains transaction logs. Use this option with the -n option.
-n *filename*	Specifies the output file that holds the SQL statements when you run the dbtran utility against a database server.
-o *filename*	Writes output messages to the named file.

Option	Description
-r	Removes any transactions that were not committed. This is the default behavior.
-rsu *username,...*	Specifies a comma-separated list of user names to override the default Replication Server user names. By default, the -is option assumes the default Replication Server user names of dbmaint and sa.
-s	Controls how UPDATE statements are generated. If you do not use this option, and there is no primary key or unique index on a table, dbtran generates UPDATE statements with a nonstandard FIRST keyword in case of duplicate rows. If you do use this option, the FIRST keyword is omitted for compatibility with the SQL standard.
-sr	Places generated comments in the output file describing how SQL Remote distributes operations to remote sites.
-t	Controls whether triggers are included in the command file. By default, actions performed by triggers are not included in the command file. If the matching trigger is in the database, when the command file is run against the database, the trigger performs the actions automatically. Trigger actions should be included if the matching trigger does not exist in the database against which the command file is to run.
-u *userid,...*	Limits the output from the transaction log to include only specified users.
-x *userid,...*	Limits the output from the transaction log to exclude specified users.
-y	Replaces existing command files without prompting you for confirmation. If you specify -q, you must also specify -y or the operation fails.
transaction-log	Specifies the log file to be translated. Cannot be used together with -c or -m options
SQL-file	Names the output file containing the translated information. For use with *transaction-log* only.

Usage

dbtran takes the information in a transaction log and places it as a set of SQL statements and comments into an output file. The utility can be run in the following ways:

- **Against a database server** When you run dbtran against a database server, the utility is a standard client application. It connects to the database server using the connection string specified following the -c option, and places output in a file specified with the -n option. DBA authority is required to run in this way.

  For example, this command translates log information from the server iqdemo and places the output in a file named *iqdemo.sql*:

  ```
  dbtran -c
  "eng=iqdemo;dbn=iqdemo;dbf=iqdemo.db;uid=DBA;pwd=sql"
  -n iqdemo.sql
  ```

- **Against a transaction log file** When you run dbtran against a transaction log, the utility acts directly against a transaction log file. Protect your transaction log file from general access to prevent users from running this statement.

  ```
      dbtran iqdemo.log iqdemo.sql
  ```

When the *dbtran* utility runs, it displays the earliest log offset in the transaction log. This can be an effective method for determining the order in which multiple log files were generated.

If -c is used, dbtran attempts to translate the online transaction log file, and all the offline transaction log files in the same directory as the online transaction log file. If the directory contains transaction log files for more than one database, dbtran may give an error. To avoid this, ensure that each directory contains transaction log files for only one database.

A transaction can span multiple transaction logs. If transaction log files contain transactions that span logs, translating a single transaction log file (for example, dbtran demo.log) can cause the spanning transactions to be lost. In order for dbtran to generate complete transactions, use the -c or -m options with the transaction log files in the directory.

Exit codes are 0 (success) or nonzero (failure).

This utility accepts @filename parameters. See “Server command-line switches” on page 8.

Ping utility (dbping)

The command-line Ping utility dbping assists in diagnosing connection problems.

Syntax

dbping [*options*]

Parameters

Table 3-7 lists the available options for dbping.

Table 3-7: dbping options

Option	Description
-c *"keyword=value; ..."*	Supply database connection parameters.
-d	Make a database connection if the server is found.
-en	Specifies that you want dbping to exit with a failed return code when NULL is returned for any of the properties specified. By default, dbping prints NULL when the value for a property specified by -pc, -pd, or -ps is unknown, and exits with a success return code. This option can only be used in conjunction with -pc, -pd, and -ps.
-l *library*	Use the specified ODBC driver or driver manager library.
-m	Use the ODBC Driver Manager. Otherwise, connect using Embedded SQL™.
-o *filename*	Log output messages to a file.
-pc *property,…*	Report specified connection properties.
-pd *property,…*	Report specified database properties.
-ps *property,…*	Report specified database server properties.
-q	Operate quietly—do not print messages.

Option	Description
-s	Returns information about the performance of the network between the computer running dbping and the computer running the database server. Approximate connection speed, latency, and throughput are displayed. The -c option is usually required to specify the connection parameters to connect to a database on the server. You can only use dbping -s for embedded SQL connections. This option is ignored if -m or -l is also specified. By default, dbping -s loops through the requests for at least one second for each statistic it measures. A maximum of 200 connect and disconnect iterations are performed, regardless of the time they take, to avoid consuming too many resources. On slower networks, it can take several seconds to perform the minimum number of iterations for each statistic. The performance statistics are approximate, and are more accurate when both the client and server computers are fairly idle.
-st *time*	This option is the same as -s, except that it specifies the length of time, in seconds, that dbping loops through the requests for each statistic it measures. This option allows more accurate timing information to be obtained than -s.
-z	Display debugging information

Usage

Use dbping to help debug connection problems. You can enter a full or partial connection string; the utility returns a message indicating whether the attempt to locate a server or database, or to connect, was successful.

Use dbping for Embedded SQL or ODBC connections. You cannot use dbping for jConnect (TDS) connections.

Exit codes are 0 (success) or nonzero (failure).

This utility accepts @filename parameters.

Options

Connection parameters (-c) See Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*. If no connection parameters are specified, connection parameters from the SQLCONNECT environment variable are used, if set.

Make database connection (-d) Ping the database, not just the server.

If you do not supply the -d option, dbping reports success if it finds the server specified by the -c option. If you do supply the -d option, dbping reports success only if it connects to both server and database.

For example, if you have a server named blair running the iqdemo database, this command succeeds:

```
dbping -c "eng=blair;dbn=iqdemo"
```

The following command fails, with the message `Ping database failed -- specified database not found`

```
dbping -d -c "eng=blair;dbn=iqdemo"
```

Load specified library (-l) Specify the library to use (without its file extension). This option does not use the ODBC Driver Manager, and so is particularly useful on UNIX operating systems.

For example, this command directly loads the ODBC driver:

```
dbping -m -c "dsn=IQ15iqdemo" -l dbodbc11
```

Use dbping to verify connectivity with the ODBC Driver Manager on UNIX systems. Unlike dbisql or other tools, dbping allows you to explicitly test the components as you would with a third-party tool. For example:

```
dbping -m -c "dsn=dsnname" -l /<full path>/libodbc.so
```

where *libodbc.so* is the third-party ODBC driver.

Use ODBC to connect (-m) Establish a connection using ODBC. By default, dbping attempts a connection using the embedded SQL interface.

Report connection properties (-pc) Upon connection, display the specified connection properties. Supply the properties in a comma-separated list. You must specify enough connection information to establish a database connection.

For a list of connection properties, see “Connection properties ” in the SQL Anywhere documentation at *SQL Anywhere 11.0.1 > SQL Anywhere Server - Database Administration > Configuring Your Database > Connection, database, and database server properties*.

Report database properties (-pd) Upon connection, display the specified database properties. Supply the properties in a comma-separated list. You must specify enough connection information to establish a database connection .

For a list of database properties, see “Database properties ” in the SQL Anywhere documentation at *SQL Anywhere 11.0.1 > SQL Anywhere Server - Database Administration > Configuring Your Database > Connection, database, and database server properties*.

Report database server properties (-ps) Upon connection, display the specified database server properties. Supply the properties in a comma-separated list.

For a list of database server properties, see “Database server properties” in the SQL Anywhere documentation at *SQL Anywhere 11.0.1 > SQL Anywhere Server - Database Administration > Configuring Your Database > Connection, database, and database server properties*.

Log output messages to file (-o) Write output messages to the named file.

Operate quietly (-q) If dbping fails, a message always appears. If succeeds, no message appears if -q is specified.

Display debugging information (-z) This option is available only when an embedded SQL connection is being attempted. That is, it cannot be combined with -m or -l. It displays the network communication protocols used to attempt connection, and other diagnostic messages.

Server Location utility (dblocate)

The Server Location command-line utility dblocate may assist in diagnosing connection problems by locating databases on the immediate TCP/IP network.

Syntax

dblocate [*options*]

Parameters

Table 3-8 lists the options available for the dblocate utility.

Table 3-8: dblocate options

Option	Description
-d	Lists the server name and address, for each server found, followed by a comma-separated list of databases running on that server. If the list exceeds 160 characters, it is truncated and ends with an ellipsis (...).
-dn *database-name*	Lists the server name and address, for servers running a database with the specified name. If the list exceeds 160 characters, it is truncated and ends with an ellipsis (...).
-dv	Displays the server name and address, for each server found, listing each database running on that server on a separate line. The list is not truncated, so this option can be used to reveal lists that are truncated when the v option is used.

Option	Description
-n	Lists IP addresses in the output, rather than computer names. This may improve performance since looking up computer names may be slow.
-o *filename*	Writes output messages to the named file.
-p *portnumber*	Displays the server name and address only for servers using the specified TCP/IP port number. The TCP/IP port number must be between 1 and 65535.
-q	Runs in quiet mode—messages are not displayed.
-s *name*	Displays the server name and address only for servers with the specified server name. If this option is used, the -ss option should not be used (if both options are used, it is likely that no matching servers will be found).
-ss *substr*	Displays the server name and address only for servers that contain the specified substring anywhere in the server name. If this option is used, the -s option should not be used (if both options are used, it is likely that no matching servers will be found).
-v	Displays the full server name. By default, dblocate truncates database server names that are longer than 40 bytes.

Usage

dblocate locates any SQL Anywhere or Sybase IQ database servers running over TCP/IP on the immediate network. It prints a list of database servers and their addresses.

Depending on your network, the utility may take several seconds before printing its results.

Transaction Log utility (dblog)

Sybase IQ automatically handles the creation and deletion of the transaction log for a database. The command-line Transaction Log utility dblog displays or changes the name of the transaction log or transaction log mirror associated with your database. You can also use dblog to stop a database from maintaining a transaction log mirror, or start maintaining a transaction log mirror.

The database must run with a transaction log. The Sybase IQ server will not start without a transaction log. A transaction log mirror is a duplicate copy of a transaction log, maintained in tandem by the database. A transaction log mirror is not required, but Sybase recommends that you use one, especially if you do not frequently back up your IQ database.

Syntax

dblog *[options] database-file*

Parameters

The following table lists the options available for the dblog utility.

Table 3-9: dblog options

Option	Description
-ek *key*	Specify the encryption key for strongly encrypted databases directly in the command. If you have a strongly encrypted database, you must provide the encryption key to use the database or transaction log in any way. For strongly encrypted databases, you must specify either -ek or -ep, but not both. The command will fail if you do not specify the correct key for a strongly encrypted database.
-ep	Specify that you want to be prompted for the encryption key. This option causes a window to appear, in which you enter the encryption key. It provides an extra measure of security by never allowing the encryption key to be seen in clear text. For strongly encrypted databases, you must specify either -ek or -ep, but not both. The command will fail if you do not specify the correct key for a strongly encrypted database.
-g *n*	Use this option if you are using the Log Transfer Manager to participate in a Replication Server installation. It can be used after a backup is restored, to set the generation number.
-il	Use this option if you have stopped using the Log Transfer Manager to participate in a Replication Server installation on this database, but continue to use SQL Remote or MobiLink synchronization. It resets the Log Transfer Manager log offset that is kept for the delete_old_logs option, allowing transaction logs to be deleted when they are no longer needed.
-is	Use this option if you have stopped using MobiLink synchronization on this database, but continue to use the Log Transfer Manager or SQL Remote. It resets the MobiLink log offset that is kept for the delete_old_logs option, allowing transaction logs to be deleted when they are no longer needed.
-m *mirror-name*	Specify the file name for a new transaction log mirror. If the database is not currently using a transaction log mirror, it starts using one. If the database is already using a transaction log mirror, it changes to using the new file as its transaction log mirror.

Option	Description
-n	Stop using a transaction log, and stop using a transaction log mirror. Without a transaction log, the database can no longer participate in data replication or use the transaction log in data recovery. If a SQL Remote, Log Transfer Manager, or dbmlsync truncation offset exists, the transaction log cannot be removed unless the corresponding ignore option (-il for the Log Transfer Manager, -ir for SQL Remote, or -is for dbmlsync) is also specified. You cannot stop using a transaction log if the database has auditing turned on (unless you first turn auditing off).
-o *filename*	Write output messages to the named file.
-q	Run in quiet mode—messages are not displayed.
-r	Maintain a single transaction log for databases that maintain a transaction log mirror.
-t *log-name*	Specify the file name for a new transaction log. If the database is not currently using a transaction log, it starts using one. If the database is already using a transaction log, it changes to using the new file as its transaction log.
-x *n*	Reset the transaction log current relative offset to n, so that the database can take part in replication. This option is used for reloading SQL Remote consolidated databases.
-z *n*	Reset the transaction log starting offset to n, so that the database can take part in replication. This option is used for reloading SQL Remote consolidated databases.

Usage

The dblog command line utility allows you to display or change the name of the transaction log or transaction log mirror. You can also stop or start maintaining a transaction log mirror.

The name of the transaction log is set when you create a database. The database cannot be running when you change its transaction log file name.

When you use the RESTORE statement to move and/or rename a database, you can rename all of the files except the transaction log. Transactions continue to be written to the old log file name, in the location where the catalog store file (the *.db* file) is located after the database is restored.

When you rename or move all other files in the database, Sybase recommends that you do the same for the log file. Use dblog to move or rename the log file. Run this utility after using RESTORE with:

- A new database name
- The RENAME option

You can use dblog to rename the transaction log even if you have not restored the database, given these restrictions:

- The IQ server must be stopped.
- The databases cannot be participating in SQL Remote replication. For Sybase IQ, this means that dblog cannot be used in multiplex environments.
- After the log is renamed, retain the old log until the next database backup, in case the old log is needed for recovery from a media failure.

dblog displays additional information about the transaction log:

- Version number
- Starting offset, for use in replication
- Ending offset, for use in replication
- Page size
- Total number of pages
- Number of empty pages
- Percentage of the log file in use

See “The transaction log file” in Chapter 4, “Managing System Resources” in the *Performance and Tuning Guide*.

Validation utility (dbvalid)

The command-line Validation utility dbvalid validates the indexes and keys on some or all of the SQL Anywhere tables in the catalog store. The Validation utility scans the entire table and looks up each record in every index and key defined on the table. By default, the Validation utility uses the express check option.

Note The dbvalid utility lets you easily validate SQL Anywhere catalog store tables, but does not validate IQ tables. Use the IQ stored procedure sp_iqcheckdb to validate IQ tables.

You can access the dbvalid utility at the system command-line level, which is useful for incorporating dbvalid into batch or command files.

Syntax

dbvalid [*options*] [*object-name*,...]

Parameters

Table 3-10 lists the options available for the dbvalid utility.

Table 3-10: dbvalid options

Option	Description
object-name	The name of a table or (if -i is used) an index to validate
-c "*keyword=value*; ..."	Supply database connection parameters
-o *filename*	Log output messages to a file
-f	Validate tables with full check
-fd	Validate tables with data check
-fi	Validate tables with index check
-fx	Validate tables with express check
-i	Each *object-name* is an index
-q	Operate quietly—do not print messages
-s	Validate database pages using checksums
-t	Each *object-name* is a table

Usage

With the dbvalid command-line utility, you can validate the indexes and keys on some or all of the SQL Anywhere tables in the catalog store. dbvalid scans the entire table and confirms that each row exists in the appropriate indexes. It is the same as running the SQL Anywhere VALIDATE TABLE statement on each catalog store table.

Note VALIDATE TABLE is not supported in Sybase IQ. sp_iqcheckdb provides a similar function for IQ store tables.

By default, the Validation utility uses the express check option. However, the express check option is *not* used if you specify -f, -fd, -fi, -fn, or -i.

If the catalog store table is inconsistent, dbvalid reports an error. If errors are reported, you can drop all of the indexes and keys on a table and re-create them. You must also recreate any foreign keys to the table.

Warning! Validate a table or entire catalog store only when no connections are making changes to the database; otherwise, spurious errors may be reported, indicating some form of database inconsistency even though no inconsistency actually exists.

Program exit code	Description
0	Database validated successfully
1	General failure in utility
2	Error validating database
7	Cannot find database to connect to (database name is wrong)
8	Cannot connect to database (user ID/password is wrong)
11	Cannot find server to connect to (server name is wrong)
12	Incorrect encryption key for starting database

Options

Connection parameters (-c) Supply database connection parameters. See “Connection parameters” in Chapter 4, “Connection and Communication Parameters” in the *System Administration Guide: Volume 1*. The user ID must have DBA authority or REMOTE DBA authority.

For example, the following validates the iqdemo database, connecting as user DBA with password sql:

```
dbvalid -c “uid=DBA;pwd=sql;dbf-c:\sybase\IQ-15_1\demo\iqdemo.db”
```

Full check for each table (-f) In addition to the default validation checks, carry out both data checks (-fd) and index checks (-fi). This option corresponds to the WITH FULL CHECK option of the SQL Anywhere VALIDATE TABLE statement. Depending on the contents of your catalog store, this option may significantly extend the time required to validate.

Data check for each table (-fd) In addition to the default validation checks, check that all of each LONG BINARY, LONG VARCHAR, TEXT or IMAGE data type can be read. Entries with these data types may span more than one page. In the IQ catalog store:

- Domain – user-defined data type.
- IMAGE – a domain to LONG BINARY.
- TEXT – a domain to LONG VARCHAR.

This option instructs the database server to check all pages used by each entry. This corresponds to the WITH DATA CHECK option on the SQL Anywhere VALIDATE TABLE statement. Depending on the contents of your catalog store, this option may significantly extend the time required to validate.

Index check for each table (-fi) In addition to the default validation checks, validate each index on the table. This corresponds to the WITH INDEX CHECK option of the SQL Anywhere VALIDATE TABLE statement. Depending on the contents of your catalog store, this option may significantly extend the time required to validate.

Express check for each table (-fx) In addition to the default and data checks, check that the number of rows in the table matches the number of entries in the index. This corresponds to the WITH EXPRESS CHECK of the SQL Anywhere VALIDATE TABLE statement. This option does not perform individual index lookups for each row.

Validate specified indexes (-i) Instead of validating tables, validate indexes. Ensure that every row referenced in the index actually exists in the table. For foreign-key indexes, -i also ensures that the corresponding row exists in the primary table. If you supply a *table-name* instead of an *index-name*, validates the primary key index. In this case, for dbvalid, each of the *object-name* values supplied represents an index instead of a table and has a name in the following format:

```
[ [ owner.]table-name.]index-name
```

Must be the owner of the table on which the index is created, have DBA authority, or have REMOTE DBA authority.

Log output messages to file (-o) Write output messages to the named file.

Operate quietly (-q) Do not display output messages.

Validate database using page checksums (-s) Checksums are used to determine whether a database page has been modified on disk. If you created a database with checksums enabled, you can validate the catalog store using checksums. Checksum validation reads each page of the catalog store from disk and calculates its checksum. If the calculated checksum is different from the checksum stored on the page, the page has been modified on disk and an error is returned. The page numbers of any invalid catalog store pages appear in the server messages window. You cannot use -s in conjunction with -i, -t, or any of the -f options.

Validate tables (-t) The list of *object-name* values is a list of tables. This is also the default behavior.

Example

The following command validates the catalog store of the iqdemo database, connecting as user DBA with password sql:

```
dbvalid -c “uid=DBA;pwd=sql;dbf-c:\sybase\IQ-15_1\demo\iqdemo.db”
```

See also

- “sa_validate system procedure” in Chapter 7, “System Procedures” in *Reference: Building Blocks, Tables, and Procedures*
- “sp_iqcheckdb procedure” in Chapter 7, “System Procedures” in *Reference: Building Blocks, Tables, and Procedures*
- “Database verification” in Chapter 13, “System Recovery and Database Repair” in the *System Administration Guide: Volume 2*

CHAPTER 4 The SQL Preprocessor

About this chapter

This chapter provides reference information for the SQL preprocessor (iqsqlpp).

Contents

Introduction to the SQL preprocessor

Embedded SQL is a database programming interface for the C and C++ programming languages. Embedded SQL consists of SQL statements intermixed with (embedded in) C or C++ source code. These SQL statements are translated by a SQL preprocessor into C or C++ source code, which you then compile.

The Sybase IQ SQL preprocessor utility iqsqlpp translates the SQL statements in an input file (*.sqc*) into C language source that is put into an output file (*.c*).

Running the SQL preprocessor (iqsqlpp)

This section provides the syntax of the iqsqlpp command and describes the iqsqlpp command-line options.

Syntax

iqsqlpp [*options*] *<in filename >* [*<out filename>*]

Parameters

Table 4-1 lists the options available for the iqsqlpp utility.

Table 4-1: iqsqlpp options

Option	Description
-d	Favor data size.
-e *<flag>*	Flag nonconforming SQL syntax as an error, where *<flag>* is one of the following: e, i, f, t, u, w, c99, c03, p99, p03.
-h *<width>*	Limit the maximum line length of output.
-k	Include user declaration of SQLCODE.
-n	Line numbers.
-o *<O/S spec>*	Target operating system specification (WINDOWS, WINNT or UNIX).
-q	Quiet mode—do not print banner.
-r-	Generate reentrant code.
-s *<len>*	Maximum string constant length for the compiler.
-w *<flag>*	Flag nonconforming SQL syntax as a warning, where *<flag>* is one of the following: e, i, f, t, u, w, c99, c03, p99, p03.
-x	Change multibyte SQL strings to escape sequences.
-z *<cs>*	Specify the collation sequence. For a list of recommended collation sequences, enter dbinit -l at a command prompt.

Usage

The SQL preprocessor processes a C or C++ program containing Embedded SQL before the compiler is run. iqsqlpp translates the SQL statements in the input file *sql-filename* into C language source that is put into the *output-filename*. The normal extension for source programs with Embedded SQL is *.sqc*. The default output file name is the *sql-filename* with an extension of *.c*. If the *sql-filename* has a *.c* extension, the default output file name extension is *.CC*.

Options

-d Favor data size. Generate code that reduces data space size. Data structures are reused and initialized at execution time before use. This increases code size.

-e *<flag>* This option flags, as an error, any Embedded SQL that is not part of a specified set of SQL92.

The allowed values of <*flag*> and their meanings are as follows:

- **e** Flag syntax that is not entry-level SQL92 syntax.
- **i** Flag syntax that is not intermediate-level SQL92 syntax.
- **f** Flag syntax that is not full-SQL92 syntax.
- **t** Flag non-standard host variable types.
- **u** Flag syntax that is not supported by UltraLite.
- **w** Allow all supported syntax.
- **c99** Flag syntax that is not core SQL/1999 syntax.
- **c03** Flag syntax that is not core SQL/2003 syntax.
- **p99** Flag syntax that is not full SQL/1999 syntax.
- **p03** Flag syntax that is not full SQL/2003 syntax.

-h *width* Limits the maximum length of lines output by iqsqlpp to *width*. The continuation character is a backslash (\), and the minimum value of *width* is ten.

-k Notifies the preprocessor that the program to be compiled includes a user declaration of SQLCODE.

-n Generate line number information in the C file. This consists of *#line* directives in the appropriate places in the generated C code. If the compiler you are using supports the *#line* directive, this option makes the compiler report errors on line numbers in the SQC file (the file with the Embedded SQL) as opposed to reporting errors on line numbers in the C file generated by the SQL preprocessor. Also, the *#line* directives are used indirectly by the source level debugger so that you can debug while viewing the SQC source file.

-o <*O/S spec*> Specify the target operating system. This option must match the operating system where you run the program. A reference to a special symbol is generated in your program. This symbol is defined in the interface library. If you use the wrong operating system specification or the wrong library, an error is detected by the linker. The supported operating systems are:

- **WINDOWS** Microsoft Windows
- **UNIX** Use this option if you are creating a 32-bit UNIX application.
- **UNIX64** Use this option if you are creating a 64-bit UNIX application.

-q Operate quietly. Do not print the banner.

-r- Generate reentrant code. For more information on reentrant code, see “SQLCA management for multi-threaded or reentrant code” in the SQL Anywhere documentation at *SQL Anywhere 11.0.1 > SQL Anywhere Server - Programming > SQL Anywhere Data Access APIs > SQL Anywhere embedded SQL > The SQL Communication Area (SQLCA)*.

-s *<len>* Set the maximum size string that the preprocessor puts into the C file. Strings longer than this value are initialized using a list of characters (*“a,” “b,” “c,”* and so on). Most C compilers have a limit on the size of string literal they can handle. This option is used to set that upper limit. The default value is 500.

-w <*flag*> This option flags any Embedded SQL that is not part of a specified set of SQL92 as a warning.

The allowed values of <*flag*> and their meanings are as follows:

- **e** Flag syntax that is not entry-level SQL92 syntax.
- **i** Flag syntax that is not intermediate-level SQL92 syntax.
- **f** Flag syntax that is not full-SQL92 syntax.
- **t** Flag non-standard host variable types.
- **u** Flag syntax that is not supported by UltraLite.
- **w** Allow all supported syntax.
- **c99** Flag syntax that is not core SQL/1999 syntax.
- **c03** Flag syntax that is not core SQL/2003 syntax.
- **p99** Flag syntax that is not full SQL/1999 syntax.
- **p03** Flag syntax that is not full SQL/2003 syntax.

-x Change multibyte strings to escape sequences so that they can pass through compilers.

-z <*cs*> This option specifies the collation sequence. For a list of recommended collation sequences, enter dbinit -l at a command prompt.

The collation sequence helps the preprocessor understand the characters used in the source code of the program, for example, in identifying alphabetic characters suitable for use in identifiers. If you do not specify -z, the preprocessor attempts to determine a reasonable collation to use, based on the operating system and the SALANG and SACHARSET environment variables.

See also

“SQL Anywhere embedded SQL” in the SQL Anywhere documentation at *SQL Anywhere 11.0.1 > SQL Anywhere Server - Programming > SQL Anywhere Data Access APIs*.

"SQL Preprocessor messages" in *Error Messages*.

Index