SQLBase™

Starter Guide

Product Version 12

GUPTA

The software described in this book is furnished under a license agreement and may be used only in accordance with the terms of the agreement.

Last updated: July 2015

Legal Notice

Copyright © 2014-2015 Gupta Technologies, Inc. All rights reserved.

Gupta, Gupta Technologies, the Gupta logo, Gupta Powered, the Gupta Powered logo, ACCELL, Centura, Centura Ranger, the Centura logo, Centura Web Developer, Component Development Kit, Connectivity Administrator, DataServer, DBIntegrator, Development Kit, eWave, Fast Facts, NXJ, Object Nationalizer, Quest, Quest/Web, QuickObjects, RDM, Report Builder, RPT Report Writer, RPT/Web, SQL/API, SQLBase, SQLBase Exchange, SQLBase Resource Manager, SQLConsole, SQLGateway, SQLHost, SQLNetwork, SQLRouter, SQLTalk, Team Developer, Team Object Manager, TD Mobile, Velocis, VISION, Web Developer and WebNow! are trademarks of Gupta Technologies and may be registered in the United States of America and/or other countries. SQLWindows is a registered trademark and TeamWindows, ReportWindows and EditWindows are trademarks exclusively used and licensed by Gupta Technologies.

The product described in this document is distributed under licenses restricting its use, copying, distribution, and decompilation/reverse engineering. No part of this document may be reproduced in any form by any means without prior written authorization of Gupta Technologies Corporation and its licensors, if any.

THE DOCUMENTATION IS PROVIDED “AS IS” AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. GUPTA TECHNOLOGIES, INC. SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

This document may describe features and/or functionality not present in your software or your service agreement. Contact your account representative to learn more about what is available with this Gupta Technologies® product.

Gupta Technologies, Inc.
1420 Rocky Ridge Drive, Suite 380
Roseville, CA 95661

Gupta Technologies.com

Table of Contents

Chapter 1 – Introducing SQLBase

This chapter contains a general discussion of some of the features of SQLBase. It describes the two SQLBase packages, SQLBase Server and SQLBase Desktop, and their respective components.

Each topic mentioned in this book is covered in far greater detail in other books that make up the SQLBase Books Online collection. Make sure to check there for more information.

SQLBase Platforms

SQLBase is supports the following platforms:

- Windows
- Red Hat Linux (see the release notes for exact versions)

SQLBase supports multiple users with remote communication through a network as well as local communication.

SQLBase Packages

SQLBase comes with a single installation package that installs the version of SQLBase you licensed. Two versions are offered: SQLBase Desktop (single-user) and SQLBase Server (multi-user).

SQLBase Desktop

SQLBase Desktop is a single-user database server that runs on Windows. It is licensed by machine, and comes in various licensing packs; for example, 5-machine and 25-machine packs. The installation options are the following:

Component Option	Description
Desktop Server (32-bit and 64-bit)	SQLBase Server is a 32-bit database server program that runs under Windows. It can communicate both with local and remote client applications. Client software includes: • Communication libraries: Front-end programs communicate with the server through support programs called communication libraries. • Connectivity Administrator • A tool for editing your SQL.INI file • SQL/API: The application programming interface. Included with the SQL/API is sample source code you can run to learn how the SQL/API handles certain SQLBase features. • SQLTalk: A front-end interface for SQL

Component Option	Description
	• Release Notes and certification description
SQLConsole DBA Utility	SQLConsole is a database administration and monitoring tool for SQLBase servers and databases. From a single Windows desktop, SQLConsole enables you perform administrative tasks for a local SQLBase server and all the SQLBase servers on a network.
SQLBase Programmers Interface	Contains the SQL/API software and SQLBase ++. The SQL/API is a language interface that enables you develop a client application that uses SQL. You embed SQL/API functions within your C program, which enables you to use SQL without giving up the power and flexibility of either programming language. You can create and run SQL/API applications on any of the server platforms. SQLBase ++ is a C++ based source library for application development.
SQLBase ODBC Driver	A standard ODBC 3.5 driver that fully supports multi-threaded applications
SQLBase OLE DB Data Provider	An industry-standard set of COM objects providing access to SQLBase databases. Supports COM+ (MTS) distributed transactions.
SQLBase .NET Data Provider	An industry-standard set of ADO.NET Data provider providing access to SQLBase databases for development tools that use the .NET SDK Framework.
SQLBase JDBC Driver	A set of Java classes allowing access to SQLBase from Java applets and applications.

SQLBase Server for Windows

SQLBase Server is a multi-user database server that runs on all versions of Windows. SQLBase for Windows is licensed based on concurrent user connections and comes in various user bands; for example: 5-user, 50-user, and unlimited users. The installation options are the following:

Component Option	Description
Windows 32-bit and 64-bit Servers	SQLBase Server is a multi-user database server program that can communicate both with local and remote client applications. This option also includes the Connectivity Administrator and SQLBase Management Console (SMC). Also included is SQLBase Resource Manager, with support for COM+ transactions.
Client Software	The Client Software contains the following components: • Communication libraries: Front-end programs communicate with the server through support programs called communication libraries. • Connectivity Administrator (32-bit clients and servers only): A tool for editing your SQL.INI file • DBError: A utility for looking up SQLBase error messages. Provides the error text, reason for the error, and suggested solutions. This is typically used during the application development process. • SQL/API: The application programming interface. Included with the SQL/API is

Component Option	Description
	sample source code you can use to learn how to utilize the SQL/API. • SQLTalk: A front-end interface for SQL • Release notes and certification description
SQLConsole DBA utility	SQLConsole is a database administration and monitoring tool for SQLBase Servers and databases. From a single Windows desktop, SQLConsole enables you perform administrative tasks for a local SQLBase server and all the SQLBase servers on a network.
Documentation	Go to Start > Programs > Gupta > SQLBase 12.0 > Documentation
SQLBase ODBC Driver	An industry-standard ODBC 3.5 Driver, fully supporting multi-threaded applications
SQLBase OLE DB Data Provider	An industry-standard set of COM objects providing access to SQLBase databases, supports COM+ (MTS) distributed transactions
SQLBase .NET Data Provider	An industry-standard set of ADO.NET Data provider providing access to SQLBase databases for development tools that use the .NET SDK Framework
SQLBase JDBC Driver	A set of Java classes allowing access to SQLBase from Java applets and applications
C Programmer's interface	SQLBase C example programs and interface

SQLBase Server for Linux

SQLBase Server is a multi-user database server that runs on Red Hat Linux (see the Release Notes for exact versions supported). The installation options are the following:

Component Option	Description
Linux 32-bit Server	SQLBase Server is a 32-bit, multi-user database server program that can communicate both with local and remote client applications.
Client Software	The Client Software contains the following components: • DBError: A utility for looking up SQLBase error messages. Provides the error text, reason for the error, and suggested solutions. This is typically used during the application development process. • SQL/API: The application programming interface. Included with the SQL/API is sample source code you can use to learn how to use the SQL/API. The API binary files also contain communication components to handle TCP/IP communication with the database server. • SQLTalk: A character-mode front-end interface for SQL. You can run SQLTalk from the Linux command line. • Release notes and certification description
Documentation	Go to Start > Programs > Gupta > SQLBase 12.0 > Documentation

Component Option	Description
SQLBase ODBC Driver	An industry-standard ODBC 3.5 Driver, fully supporting multi-threaded applications
SQLBase JDBC Driver	A set of Java classes allowing access to SQLBase from Java applets and applications
C Programmer's interface	SQLBase C example programs and interface

Chapter 2 – Configuring SQLBase for Windows

This chapter discusses the following:

- Connectivity Administrator
- SQL.INI file

Connectivity Administrator

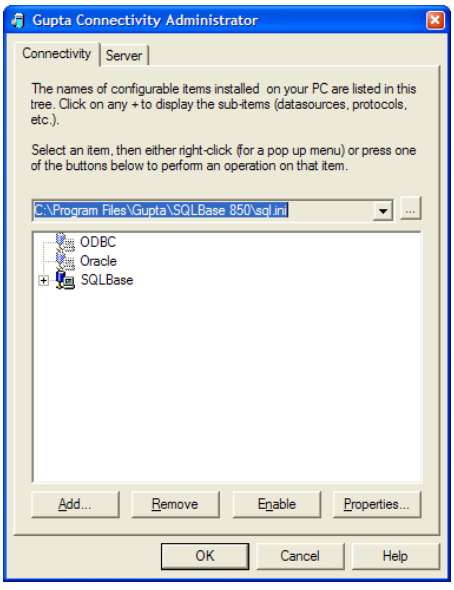
The Connectivity Administrator enables you edit your SQL.INI file, adding databases and routers. You can set the configuration for SQLBase Server, including:

- Setting the options for a SQLBase server
- Enabling and disabling communication protocols
- Adding or removing a database name
- Viewing a list of database names

The Connectivity Administrator detects all relevant software installed on your computer, including network protocols and configurations. The automated configuration also sets simple and advanced parameters of the database server. You can configure connectivity for both your client and server during the same Connectivity Administrator session. To configure your server, click the Server tab.

Using the Connectivity Administrator

1. From the Start menu, select **Programs** > **Gupta** > **SQLBase 12.0** > **Connectivity Administrator**. The following screen appears.



2. Click the **Server** tab to show the installed server.
3. Click the **+** symbol next to the server name to display listening protocols and databases.

The following actions are available when using the Connectivity Administrator:

- Left-click to select an item
- Double-click to change properties
- Right-click to display a menu of connectivity functions

For detailed information about the Connectivity Administrator, read the online help provided.

SQL.INI File Configuration Settings

Gupta products read the SQL.INI file at program start-up. The SQL.INI file is the common file containing software configuration settings. Each client and server machine must have a SQL.INI file. Beginning with version 8.5, it is possible, using Connectivity Administrator or the SQLBase Management Console, to name SQL.INI to any other name, and store it in any location on the computer. SQLBase documentation, however, always refers to this as the SQL.INI file.

Note: The Windows version of SQLBase is not case-sensitive, but the Linux version of SQLBase expects this file to be in lowercase, as follows: sql.ini.

To use a SQLBase client, you must first enable the appropriate communication libraries by setting the correct comdll value in the sql.ini file. The easiest way to do this is by setting the communications interfaces through the SQLBase Connectivity Administrator or your preferred text editor. Read the online help for the SQLBase Connectivity Administrator for more information, or refer to chapters two and three of the *Database Administrator's Guide*.

Do I need to configure sql.ini?

If you just installed SQLBase, and you want to verify that it is running correctly, you probably will not need to change the configuration in sql.ini. The installation of SQLBase for Windows provides default values that allow the server to run and to make itself available using the TCP/IP protocol on port 2155. The client components installed at the same time are configured to talk to that server and access the sample database ISLAND.

However, once you have tested SQLBase and want to begin using it for your own applications and data, you will need to make configuration changes in the sql.ini file. Reasons for this include:

- Adding new databases to the server

  The database names must be added to both the client and server portions of sql.ini.

- Restricting which databases are visible to client applications

  The default at installation time is to make all databases on a server visible.

- Changing performance settings

  sql.ini contains keywords related to cache sizes, types of joins, log files, and other performance features. You can change the values associated with those keywords.

The most complete information about configuring sql.ini is found in the *Database Administrator's Guide*, chapters 2 and 3.

How SQLBase Finds sql.ini

When a server starts or when a client attempts to connect to a server, three important configuration files must be found or an error message is issued. These are the main configuration file (sql.ini), and the error.sql and message.sql files, which influence messaging between server and client.

Logic for Servers

Each server executable accepts an optional command-line argument that contains the file name (and, optionally, the path) of the configuration file. The name might be something other than sql.ini.

An example of a command line specifying the argument is:

```
dbntsrv.exe “ini=c:\my work directory\myconfigfile.cfg”
```

If the command-line argument is left blank when the executable starts, it is presumed that the configuration file is named sql.ini and that it resides in the same directory as the executable itself.

The error.sql and message.sql files are always required to be in the same directory as the database server executable itself.

Logic for Clients

If a client application connects to the database using the API function sqliniEx, it supplies the explicit name and location of the configuration file, and the API will simply attempt to open that file. No other searching will be done. Some Gupta client tools, such as SQLTalk, can accept a command-line argument specifying the configuration file. If such an argument is specified, the sqliniEx method of connecting is used.

If no explicit configuration file is named, the SQLBase API attempts to locate a file named sql.ini in the following locations:

1. The directory named in the SQLBASE environment variable
2. The current directory
3. The \SQLBASE directory on the current drive
4. The root directory on the current drive
5. Directories specified by the PATH environment variable

Note: If the SQLBASE environment variable is set, SQLBase looks *only* in the directory to which it points. It does not continue to follow the search order outlined above.

Client Logic for error.sql and message.sql

Once the configuration file (sql.ini) is located, two other important files must be located. The search logic for error.sql and message.sql is very similar to that for sql.ini, with one important difference. Because sql.ini has already been located, we can check for the CLIENTRUNTIMEDIR keyword in that file. If the keyword exists and has a value, only the directory named by that value will be searched for error.sql and message.sql. Otherwise, the search logic shown in steps 1 through 5 above is used to locate these two files.

Running SQLBase Server as a Windows Service

To install SQLBase Server for Windows as a service, you must have Administrator privileges.

SQLBase Server for Windows offers the option, during installation, to run as a Windows service. If this option is chosen, SQLBase is set to run as a system account and to interact with the desktop.

Prior to version 8.5 only one instance of SQLBase Server could be run on a Windows machine at any given time, regardless of whether it was being run as a service or an application program. Beginning with version 8.5, multiple SQLBase Server engines can run simultaneously, although only one of these instances can be a version prior to 8.5. You can use the SQLBase Management Console (SMC) to start and stop SQLBase as a Windows service. SMC also allows you to rename a SQLBase service, and to register as a service a SQLBase instance that was previously configured to run as an application. For information, see the "DBA Operations" chapter in the *Database Administrator's Guide*.

For more information on running SQLBase Server for Windows as a service, see the "Running SQLBase Server as a Windows Service" chapter in the *Database Administrator's Guide*.

SQLBase cannot run as a service automatically if you are using the user account in the Services setup.

Chapter 3 – Configuring SQLBase for Linux

This chapter discusses configuring SQLBase on Linux machines.

sql.ini File Configuration Settings

Gupta products read the sql.ini file at program start-up. The sql.ini file is the common file containing software configuration settings. Each client and server machine must have a sql.ini file. It is possible to name sql.ini to any other name, and store it in any location on the computer. The documentation, however, always refers to this file by the name sql.ini. Because Linux is case-sensitive, any file name you select must be in lowercase.

How SQLBase Finds sql.ini

When a server starts, or when a client attempts to connect to a server, three important configuration files must be found or an error message is issued. These are the main configuration file (sql.ini) and the error.sql and message.sql files, which influence messaging between server and client.

Logic for Servers

Each server executable accepts an optional command-line argument that contains the file name (and, optionally, the path) of the configuration file. The name might be something other than sql.ini. An example of a command line specifying the argument is:

```
./dblxsrv “ini=/usr/etc/my work directory/myconfigfile.cfg”
```

If the command-line argument is left blank when the executable starts, it is presumed that the configuration file is named sql.ini and that it resides in the same directory as the executable itself.

The error.sql and message.sql files must be in the same directory as the database server executable itself.

Logic for Clients

If a client application connects to the database using the API function sqliniEx, it supplies the explicit name and location of the configuration file, and the API will simply attempt to open that file. No other searching is done. Some Gupta client tools, such as sqllxtlk, can accept a command-line argument specifying the configuration file. If such an argument is specified, the sqliniEx method of connecting is used.

If no explicit configuration file is named, the SQLBase API attempts to find a file named sql.ini in the following locations:

1. The directory named in the SQLBASE environment variable
2. The current directory
3. /SQLBASE directory on the current drive
4. The root directory on the current drive
5. Directories specified by the PATH environment variable

Note: If the SQLBASE environment variable is set, SQLBase looks only in the directory to which it points. It does not continue to follow the search order outlined above.

Client Logic for error.sql and message.sql

Once the sql.ini configuration file is located, two other important files must be located. The search logic for error.sql and message.sql is very similar to that for sql.ini, with one important difference. Because sql.ini has already been located, we can check for the CLIENTRUNTIMEDIR keyword in that file. If the keyword exists and has a value, only the directory named by that value is searched for error.sql and message.sql. Otherwise, the search logic shown in steps 1 through 5 above is used to locate these two files.

Do I need to configure sql.ini?

If you just installed SQLBase, and you want to verify that it is running correctly, you probably will not need to change the configuration in sql.,ini. The installation of SQLBase for Linux provides default values that allow the server to run and to make itself available using the TCP/IP protocol on port 2155. The client components installed at the same time are configured to talk to that server and access the sample database ISLAND. The CLIENTRUNTIMEDIR keyword in sql.ini has a value that points to the same directory in which the client program sqllxtlk is installed.

After you test SQLBase and you want to begin using it for your own applications and data, you will need to make configuration changes in the sql.ini file. Reasons for this include:

- Adding new databases to the server

  The database names must be added to both the client and server portions of sql.ini.

- Restricting which databases are visible to client applications

  The default at installation time is to make all databases on a server visible.

- Changing performance settings

  sql.ini contains keywords related to cache sizes, types of joins, log files, and other performance features. You can change the values associated with those keywords.

The most complete information about configuring the sql.ini file is located in the *Database Administrator's Guide*.

Chapter 4 – Running SQLBase

This chapter provides step-by-step instructions for starting and stopping the SQLBase Server. It also includes information on uninstalling SQLBase.

Starting and Stopping SQLBase Server for Windows

You must start the database server before clients can access a database, unless you are using the Anonymous Pipes protocol. When this protocol is used, SQLBase is started automatically when an application attempts to connect to a database on the same machine.

Starting

By default, the setup installs SQLBase as a service. If you choose to run SQLBase Server as a Windows application, you should use SQLBase Management Console or the Windows Service Manager to make sure that the SQLBase service is disabled first.

You can manually start SQLBase as a service using either SQLBase Management Console or the Windows Service Manager.

For more information on running SQLBase Server as a service program, read chapter 13 of the *Database Administrator's Guide*.

To start SQLBase as an application, select Programs, Gupta, SQLBase 11.7 and SQLBase Server from the Start menu. This item name varies according to your installation directory and the server version installed.

Stopping

It is recommended that you disconnect all users before stopping SQLBase. If you use SQLBase Management Console to stop SQLBase, that tool allows you to automatically disconnect any current users. If, however, users are still connected when you stop the server, SQLBase automatically performs recovery on these databases during the next connection. If RECOVERY is set to off, SQLBase is unable to recover the affected databases.

SQLBase as a Windows Service

SQLBase can be stopped by using either SQLBase Management Console or the Windows Service Manager. It is recommended that you use SQLBase Management Console because this tool enables you to verify that there are no users connected before you stop the server. For more information about SQLBase Management Console, see the *Database Administrator's Guide*.

SQLBase as an Application

To stop SQLBase Server, do one of the following:

- Select File, Exit from the SQLBase menu bar and then close the SQLBase Server window.

- Use the SQLBase Management Console.
- Programmatically shut down the server using either SQLTalk or using the SQLBase API. For information on the SQLBase API, see the *SQLBase SQL Application Programming Interface Reference*.

Starting and Stopping SQLBase on Linux

See the *Release Notes* for the specific Linux brands and versions supported.

Starting SQLBase

The name of the SQLBase binary is dblxsrv. Navigate to the directory that contains this binary and enter the binary name as a command. Enter ./dblxsrv from the command prompt; do not double-click the file in File Manager.

When SQLBase starts, it displays a console window showing basic statistics about the server. To learn more about this window, see chapter 5 of the *Database Administrator's Guide*.

Stopping SQLBase

In the console window that displays SQLBase statistics, press Ctrl-X. If users are currently connected, SQLBase prompts whether to stop anyway. You can answer **Y** or **N**. Otherwise, SQLBase stops immediately.

Uninstalling SQLBase

Windows

To remove the SQLBase software from your Windows computer, use the **Add/ Remove Programs** applet in the systems Control Panel.

Linux

To remove the SQLBase software from your Linux computer, navigate to the uninst subdirectory beneath the main program directory and run the following command:

```
./uninstaller.bin
```

The uninstaller runs as a GUI application. Add the -console argument to the end of the command line to run as a console application.

Chapter 5 – Final Steps

This chapter contains information on testing your configuration using SQLTalk, National Language Support, using drivers and data providers, and loading database files to the SQLBase Server.

Finishing your Configuration

You are encouraged to edit sql.ini to improve the performance of the SQLBase server and client. You can set values for the following keywords.

Server-Related Settings

- **fileaccess** should be set to 0 for security reasons.
- **cache** should be set to one-quarter of your of physical memory for good performance.
- **dbdir** should be set to a specific directory; initially, the one in which the ISLAND subdirectory can be found.
- **servername** should be set to a unique name. This unique name must be echoed in some of the client-related settings, too.
- **password** should be set to a non-null value for increased security.

Client-Related Settings

- **clientruntimedir** should be set to a directory, even if it is the same as the directory that contains your client components.
- **serverpath** should be configured with correct information for the servers that your client will be communicating with.

For full details on the meaning and possible values of each keyword, see chapter 3 of the *Database Administrator's Guide.*

Testing your Installation

You can test the connection between your server and client using SQLTalk or SQLConsole.

SQLTalk

Before starting SQLTalk, make sure that the server program is running, as described in the previous chapter.

Starting the SQLTalk Program in Windows

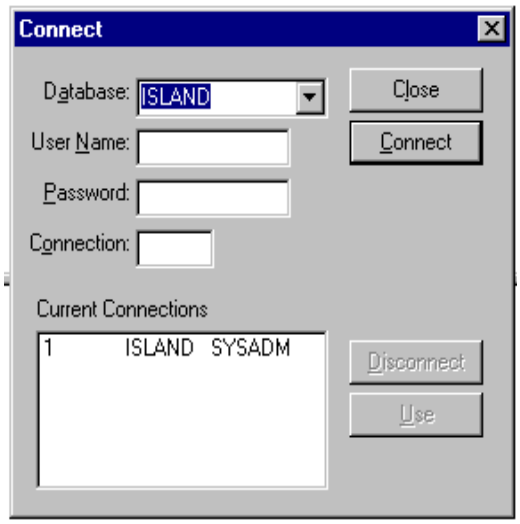
To start SQLTalk, invoke the SQLTalk item from the Gupta program group.

Using SQLTalk with Windows

You can run SQLTalk using the sqltalk.exe Windows interface. The sqlnttlk.exe that runs on Windows behaves much like the Linux version described in the section that follows.

1. Make sure that the server program is running (see the “Starting and Stopping the Server” section). On some single-user installations in Windows, the server starts automatically when a client program like SQLTalk begins, but ordinarily the server must be started in advance.
2. Start SQLTalk on the client machine, as described in the previous section. Select **Connect** from the Session menu.
3. In the dialog box, enter a database name; for example, ISLAND (the default database that ships with SQLBase) and click **Connect**.

   If your network and software components are configured correctly, a connection is added to the list as shown in the following Windows example:



   The Current Connections box displays ISLAND as the database name, which confirms your connection to this database.

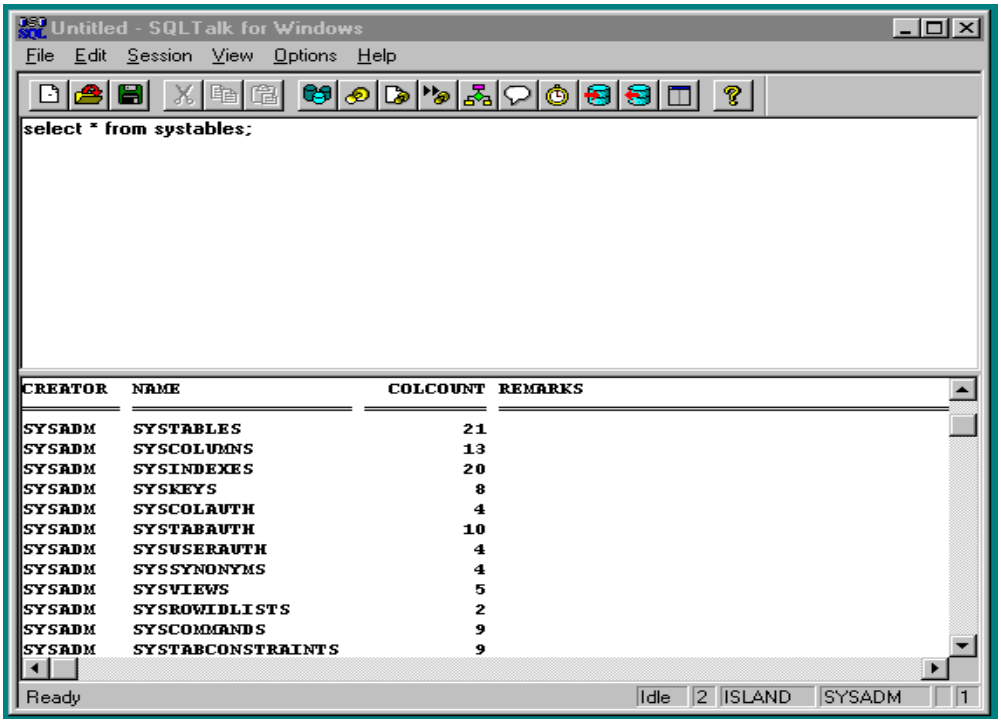
   Click **Close**.

   As an alternate technique in Windows, you can also connect to a database by entering a connection command in a SQLTalk window; for example:

```
CONNECT ISLAND;

CURSOR 1 CONNECTED TO ISLAND
```

4. Enter SQLTalk commands. Remember to end every command with a semicolon. You can start entering commands at the cursor in the top pane of the SQLTalk window; for example:



To execute the command that the cursor is currently over, press **Ctrl+Enter**.

Starting the SQLTalk Program in Linux

The SQLTalk executable is named sqllxtlk. Navigate to the directory in which it is contained (or make sure the directory is in the path) and launch the executable.

Using SQLTalk on Linux

Entering commands in the Linux version of SQLTalk is similar to Windows, but the console interface is more basic. All commands are ended with a semicolon. The following is an example of a sign-on and simple query:

```
SQLTALK Interactive Data Manager (Version 9.0.0)
Copyright (c) Gupta Technologies LLC 1985-04. All Rights Reserved.

Database[DEMO    ]: ISLAND

Username[SYSADM  ]: SYSADM

Password[********]:


Connecting to ISLAND in Interactive Line Mode


Enter a SQL command

SQL> select * from product;

   STYLE_ID STYLE
=========== ==============================
          1 Five-O Tourist Shirt
          2 Hang-Ten Surf Pants
          3 Hawaiian Mu Mu
          4 Kona Khakis
          5 Lanai Lingerie
          6 Maui Tank Top
          7 North Shore Swimsuit

7 ROWS SELECTED

Enter a SQL command

SQL>
```

You can enter SQL and SQLTalk commands as described in the *SQL Language Reference Manual* and the *SQLTalk Reference Manual.*

Ending SQLTalk

When you are ready to end your SQLTalk session, enter this command at the cursor:

```
exit;
```

On Linux, you can also press **Ctrl-C** to end the session. On Windows, you can press **Alt-F4**.

National Language Support

SQLBase supports English as its standard language and also many international languages including those spoken in Europe and Asia. For information on creating databases that support languages other than English, see the “National Language Support” chapter in the *Database Administrator’s Guide*. To understand your options when using NLS, see the *Database Administrator’s Guide* section on SQL.INI keyword LimitNLSCompares.

Using Drivers and Data Providers

SQLBase offers five main methods for clients to connect to the server:

- Native API (for Windows and Linux)
- ODBC driver (for Windows and Linux)

- JDBC driver (for Windows and Linux)
- OLE DB data provider (Windows only)
- .NET data provider (Windows only)

The first method is discussed in the *SQLBase API Reference*. For detailed information on the other four methods, see *Connecting to SQLBase*.

Client Applications

All client applications that connect to SQLBase require a configuration file (sql.ini). Even clients that are simply using one of the drivers or data providers need a configuration file, with the exception of clients using the JDBC driver. For information on how to structure the configuration file, see chapters 2 and 3 of the *Database Administrator's Guide*.

Linux Considerations

If you run a Linux client application from a directory other than the one where the binary is actually located, and you did not specify the clientruntimedir keyword in your configuration file, you might encounter an error due to the program's inability to find the supporting binary files. To remedy this, you can either change to the binary's directory before running, or you can use the LD_LIBRARY_PATH environment variable to point to the directory that contains those binary files. You would do this with commands similar to the foolowing example:

```
$LD_LIBRARY_PATH=SQLBaseBinaryDirectory
$export LD_LIBRARY_PATH
```

Loading Database Files into SQLBase

If you are upgrading from a previous version of SQLBase, it is recommended that you unload your old database files with your old version of SQLBase client and then load them to the new SQLBase Server. There is an auto-migration feature, but it is subject to several limitations. For more information, see the release notes.

You can also load a non-SQLBase database to SQLBase using a common file format, such as .CSV. For information on loading and unloading, read the LOAD and UNLOAD command documentation in the *SQL Language Reference*.

You can improve the performance of the load by performing the following tasks:

1. Defragment your disk. This is also useful for server operations in general.
2. Run SET RECOVERY OFF before you run LOAD. Be aware that when you set recovery off, SQLBase does not generate log files and this speeds up the load. When you are finished with the load, run SET RECOVERY ON.
3. Run the LOCK DATABASE command to place an exclusive lock on the database. When you are finished with the load, run UNLOCK DATABASE.
4. Use the ON SERVER clause of the LOAD and UNLOAD command. This decreases network traffic by performing the operation on the server machine, if the load file is local to the server.

5. Set the server screen to show the minimal level of process detail (level 0). This is also useful for server operations in general.