

---

*StarSQL™ for UNIX*  
*User's Guide*

*Version 5.5*

**STARQUEST**

---

---

### **Statement of Limitations on Warranty & Liability**

StarQuest Ventures, Inc. makes no representations or warranties about the suitability of the software and documentation, either expressed or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. StarQuest Ventures shall not be liable for any damages suffered by licensee as a result of using, modifying, or distributing this software or its derivatives.

StarSQL™ is a trademark of StarQuest Ventures, Inc. All trademarks or registered trademarks are the property of their respective owners.

© Copyright 1996–2009 by StarQuest Ventures, Inc.  
All rights reserved.

# Contents

<b>Introduction</b> .....	<b>9</b>
The StarSQL for UNIX Driver .....	9
The ODBC Driver Manager .....	10
Network Requirements .....	12
Network Protocols .....	12
Supported DB2 Host Systems .....	12
Data Type Mappings .....	13
Documentation .....	15
Quick Path to Using StarSQL .....	15
StarSQL Product Documentation .....	15
User's Guide .....	15
Release Notes .....	16
Command Manual Pages .....	17
Third-Party Information Resources .....	17
Contacting StarQuest .....	20
Support .....	20
Sales and Service .....	21
<b>Installing 32-bit StarSQL for UNIX</b> .....	<b>23</b>
Client Computer Requirements .....	23
Upgrading StarSQL .....	24
StarSQL for UNIX 32-bit Driver Installation .....	25
Removing the StarSQL 32-bit Driver .....	26
<b>Installing 64-bit StarSQL for UNIX</b> .....	<b>27</b>
Client Computer Requirements .....	27
Determining Application Architecture .....	28
Upgrading StarSQL .....	28
StarSQL for UNIX 64-bit Driver Installation .....	28
Removing StarSQL from a 64-bit Computer .....	29

<b>Using StarSQL for UNIX</b> .....	<b>31</b>
Licensing StarQuest Products .....	31
Node-Locked License .....	31
Floating License .....	32
Configuring a StarSQL License .....	32
Customizing the unixODBC Driver Manager Configuration .....	34
Configuring Data Sources .....	36
Testing the StarSQL/DB2 Connection .....	40
Testing the Network Connection .....	40
Testing the unixODBC Driver Manager Connection .....	40
Using the isql Utility to Test Connections .....	41
Using StarSQL with ODBC Applications .....	42
Developing ODBC Applications for StarSQL .....	42
<b>Preparing Hosts for StarSQL Access</b> .....	<b>45</b>
Preparation Required for All Hosts .....	45
User Accounts .....	46
Permissions .....	47
Preparing DB2 on an OS/390 Host .....	47
Configuring DDF .....	47
Starting DDF .....	48
Supporting Password Management Using DRDA Flows .....	49
Using StarSQL with Stored Procedures .....	49
Registering Stored Procedures .....	49
Calling Stored Procedures .....	51
Preparing a DB2 for i Host .....	51
Creating a Library for SQL Packages .....	51
Determining the RDB Name .....	52
Enabling DRDA Over TCP/IP .....	52
Using StarSQL Stored Procedure for Password Management .....	53
Registering Stored Procedures on AS/400 .....	54
Considerations for Specific AS/400 Releases .....	54
OS/400 V6R1 and Later Issues .....	54
OS/400 V5 Issues .....	54
AS/400 V4R4 Issues .....	56

Preparing a DB2 UDB Host .....	57
Enabling DRDA Support for TCP/IP .....	57
Using DB2 Control Center to Specify the DRDA Port .....	57
Using db2 Commands to Specify the DRDA Port .....	58
Enabling Encryption .....	58
Using the DB2 Control Center to Enable Encryption .....	59
Using db2 Commands to Enable Encryption .....	59
Locating the Database Name .....	59
Preparing a DB2 Server for VSE & VM .....	60
<b>Binding Packages .....</b>	<b>61</b>
Compatibility Among SQL Catalog Packages .....	61
Binding Packages with the Explain Option Enabled .....	62
Catalog Package .....	63
Dynamic SQL Packages .....	63
Static SQL Packages .....	66
Permissions for Packages .....	66
Static and Catalog Packages .....	66
Dynamic SQL Packages .....	66
Granting Use Permissions .....	67
For DB2 for OS/390 .....	67
For DB2 UDB (Windows and UNIX) .....	67
For DB2 for i .....	68
For VM and VSE .....	68
AutoBind Option .....	68
<b>Manually Installing 32-bit StarSQL for UNIX .....</b>	<b>71</b>
Installing StarSQL on AIX .....	71
Installation Using SMIT .....	71
Installation Using a Character-Oriented Terminal .....	72
Installing StarSQL on FreeBSD .....	73
Installing StarSQL on HP-UX .....	73
Using <b>swinstall</b> in Interactive Mode .....	74
Using <b>swinstall</b> in Non-Interactive Mode .....	74
Installing StarSQL on Linux .....	75
RPM Installation .....	75
Installation Using tar .....	76

Installing StarSQL on Solaris . . . . .	77
Installation Using X Windows . . . . .	77
Installation Using a Character-Oriented Terminal . . . . .	78
<b>Manually Installing 64-bit StarSQL for Linux . . . . .</b>	<b>79</b>
Installing StarSQL on a Linux Computer . . . . .	79
Installation of StarSQL Using RPM . . . . .	79
Installation of StarSQL Using tar . . . . .	80
<b>StarSQL for UNIX Command Reference . . . . .</b>	<b>81</b>
pkgviewer . . . . .	82
restart . . . . .	84
simpleconn . . . . .	86
starlic-clientcfg . . . . .	88
starping . . . . .	90
trcstart . . . . .	91
trcviewer . . . . .	92
<b>Customizing the StarSQL for UNIX Configuration . . . . .</b>	<b>93</b>
Setting Global DSN Parameters . . . . .	93
Global DSN Keywords . . . . .	94
AlwUpd . . . . .	94
AutoTypDefOvr . . . . .	94
CacheUID . . . . .	95
Explain . . . . .	95
Setting Up System and User DSN Files . . . . .	95
Sample DSN File . . . . .	97
System and User DSN Keywords . . . . .	99
Accounting . . . . .	99
AllowSynonyms . . . . .	100
AutoBind . . . . .	100
AutoSqlSet . . . . .	101
AutoTypDefOvr . . . . .	101
BinaryCCSID . . . . .	102
BindRules . . . . .	102
CacheUID . . . . .	103
Capture . . . . .	103
CaptureDDL . . . . .	103

CaptureQryOnly . . . . .	103
CatFilters . . . . .	103
CatQual . . . . .	104
CharacterSubstituion . . . . .	104
CmdBufSiz . . . . .	104
CreateTable . . . . .	104
CustomizePrdid . . . . .	105
DefaultQualifier . . . . .	106
Description . . . . .	106
ExpirationWarning . . . . .	106
FetchAhead . . . . .	106
FoldUIDPWD . . . . .	106
GetInfoCatalogUsage . . . . .	106
HeldCursors . . . . .	107
IncludeSynonyms . . . . .	108
IsolationLevel . . . . .	108
KeepDynamic . . . . .	109
LiteralConversion . . . . .	109
LongStrParams . . . . .	109
MaxRows . . . . .	110
OverrideCodeset . . . . .	110
PackageFile . . . . .	110
ParameterMarkersOnly . . . . .	111
PasswordProc . . . . .	111
PkgOwnID . . . . .	111
QryBufSiz . . . . .	111
ReadOnly . . . . .	112
SpecialColumns . . . . .	112
StrictParsing . . . . .	112
TypDefOvr . . . . .	112
UseDSCRDBTBL . . . . .	113
UseDynamicCatalogSQL . . . . .	113
UseEncryption . . . . .	113
UseJumboPackages . . . . .	114
UseStaticMatch . . . . .	114
UseSYSDUMMYAEU . . . . .	115

<b>StarLicense Client Configuration Utility</b> .....	<b>117</b>
Adding a Connection to a StarLicense Server .....	118
Removing a StarLicense Server Definition .....	118
Testing License Checkout .....	119
Displaying the StarLicense Configuration .....	119
Setting the StarLicense Client Logging Level .....	120
<b>StarSQL National Language Support</b> .....	<b>121</b>
Character Encoding of Data Across Systems .....	122
StarSQL and Unicode .....	122
Determining and Setting the Client Code Page .....	123
Determining the Host Code Page .....	123
StarSQL Implementation Details .....	123
Customizing StarSQL for National Language Support .....	124
Connecting an MBCS Client to an SBCS Host. ....	125
Supporting Asian Languages with Extended EBCDIC. ....	125
Currently Supported CCSIDs .....	126
<b>Glossary</b> .....	<b>127</b>
<b>Index</b> .....	<b>131</b>



StarSQL is available as an ODBC driver for Windows- and UNIX-based computers, and as a JDBC driver for any computer that has the Java Runtime Engine (JRE) or Java Virtual Machine (JVM) installed.

The StarSQL software is not copy protected, rather the usage is limited based upon the maximum number of concurrent connections (“CCs”) licensed. The StarQuest license allows for you to install and run any of the StarSQL drivers on any number of client computers, subject to terms of the license grant. CCs may be made available for clients on a single computer (“Node-locked License”) or any computer on the network (“Floating License”).

## The StarSQL for UNIX Driver

The StarSQL for UNIX driver is an ANSI driver that is compliant with the ODBC v3.0 specification. The StarSQL driver resides on the UNIX client computer, and the data resides in a DB2 database on a host computer.

The StarSQL driver performs the following functions to enable ODBC applications to communicate with a DB2 host:

- provides a programmatic interface for executing dynamic SQL functions
- provides transparent translation of character encoding between different client and host operating systems

StarSQL for UNIX supports directly connecting to a DB2 host in a TCP/IP network. To access a DB2 host on an SNA network a StarSQL for UNIX client can use the StarPipes for Windows service, which is a separate product available from StarQuest. Refer to the [StarPipes product page](#) on the StarQuest web site, or contact StarQuest Customer Support (see [page 20](#)), if you need to support UNIX clients in an SNA network.

The v5.5 release of StarSQL for UNIX contains two versions of the ODBC driver—a 32-bit version and a 64-bit version. The 32-bit version can be installed and run on either a 32-bit or 64-bit computer; the 64-bit version is optimized for use only on a 64-bit computer. Note that V5.5 adds support for x64 StarSQL for UNIX on Linux platforms. Additional platforms such as Sun Solaris, HP-UX, IBM AIX and FreeBSD will be supported with x64-bit versions in subsequent releases.

In addition to the 64-bit version of the driver, the StarSQL for UNIX v5.5 release includes the following significant enhancements:

- a graphical interface for binding SQL packages on the host
- inclusion of the unixODBC driver manager graphical interface for configuring data sources, ODBCConfig
- simplified installation and configuration, eliminating the need to set environment variables
- support for system data sources, which can be shared among users

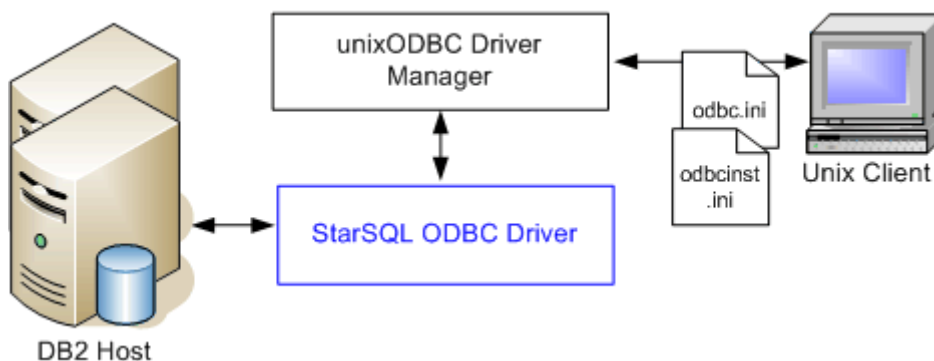
This User's Guide describes how to install, configure, and use both the 32-bit and 64-bit version of the StarSQL for UNIX driver. Refer to the product documentation that is included with the StarSQL for Java or StarSQL for Windows driver to install, configure, and use those versions of the driver.

## The ODBC Driver Manager

An ODBC driver manager works with an ODBC driver to provide inter-operability among multiple types of databases (such as DB2 and Oracle), and runtime binding to a data source. This allows an application to link only with the driver manager shared object (such as `libodbc.so` on Linux), eliminating dependencies on which ODBC driver will be used. StarSQL for UNIX includes the unixODBC driver manager for use with the StarSQL driver.

The ODBC Driver Manager maintains a repository of the installed ODBC drivers, including the StarSQL driver, in a file named `odbcinst.ini`. The configuration information for system data sources, including those used by the StarSQL driver, are stored in a file named `odbc.ini`. As shown in the following illustration, the ODBC Driver Manager uses the information in the driver repository and the data source configuration to load the appropriate driver. The StarSQL driver manages the data that is sent between the client and host computers.

**Figure 1. StarSQL Driver and ODBC Driver Manager Enable UNIX Client to Communicate with DB2 Host**



The ODBC driver manager resolves data source names and ensures that the proper driver is loaded and unloaded at the appropriate time. The driver manager also maps calls from applications developed to the ODBC v2 API to the ODBC v3 API so the applications based on v2 can run under the v3 standard without modification.

The StarSQL for UNIX distribution includes the unixODBC driver manager, which is an open source project available under the GNU Library General Public License (LGPL). For more information about unixODBC, see <http://www.unixodbc.org>.

If an existing driver manager that meets the minimum requirements is not found when you install the StarSQL software, the unixODBC driver manager that is included with StarSQL is installed. The StarSQL distribution also includes the driver manager ODBCCfg utility to help you easily configure StarSQL data sources for an ODBC application to use.

---

### Note

The version of unixODBC that is included with StarSQL meets the minimum requirements and has been tested to work with the StarSQL driver. The StarSQL driver requires the following versions of the unixODBC driver manager:

- Release 2.2.12 or later of the 32-bit unixODBC driver manager to use the 32-bit version of the StarSQL driver.
  - Release 2.2.14 or later of the 64-bit unixODBC driver manager on a 64-bit computer.
-

## Network Requirements

The network requirements for using StarSQL and a list of the supported DB2 hosts are provided below. The system requirements for the client computer are provided on [page 23](#) for the 32-bit version of StarSQL, and on [page 27](#) for the 64-bit version. Because the StarSQL configuration involves the client computer, the network, and the host, several individuals may be involved with setting up the StarSQL environment.

### Network Protocols

StarSQL for UNIX supports TCP/IP access to a DB2 host. This can be provided either via direct TCP/IP access to a DB2 system that supports DRDA over TCP/IP, or via a connection through a StarPipes gateway to a DB2 host on an SNA network.

### Supported DB2 Host Systems

Before using StarSQL on the UNIX computer, the host must be prepared as described in "[Preparing Hosts for StarSQL Access](#)" on [page 45](#). You will need information about the network and the host databases, as described in "[Preparation Required for All Hosts](#)" on [page 45](#), to configure StarSQL data sources after StarSQL is installed.

StarSQL for UNIX can connect to any of the following host databases:

- DB2 for OS/390 and z/OS v5.1 and later
- DB2 for i (formerly known as DB2/400, DB2 UDB for iSeries, and DB2 for i5/OS) running OS/400 V3R2 and later
- DB2 Universal Database (UDB) for Linux, UNIX, and Windows v6.1 and later
- DB2 for VM/VSE (formerly known as SQL/DS) v3.3 and later
- DB2 Server for VSE & VM v7.1 and later

The following host databases provide native DRDA over TCP/IP support to StarSQL clients:

- DB2 for OS/390 and z/OS v5.1 and later
- DB2 for i, running OS/400 V4R2 and later
- DB2 UDB for Linux, UNIX, and Windows v6.1 and later
- DB2 Server for VSE & VM v7.1 and later

To take advantage of the StarSQL for UNIX support for password encryption and Large Object (LOB) data types, the database system also must support these advanced features.

The following database versions support LOB data types. Be sure to install all LOB-related fixes to your host system.

- DB2 for OS/390 and z/OS v6.1 and later
- DB2 for i, running OS/400 V5R1 and later with appropriate PTFs  
To support LOB data types on an OS/400 V5R1 host, see "[V5R1 Considerations](#)" on page 56.
- DB2 UDB for Linux, UNIX, and Windows v8.1 and later

The following database systems support or successfully negotiate the use of password encryption:

- DB2 OS/390 and z/OS v6.1 and later
- DB2 for i, running OS/400 V5R1 and later  
To support password encryption on a DB2 UDB for iSeries V5R1 or V5R2 host, you must install the Cryptographic Access Provider 56-bit for iSeries (#5722-AC2) or Cryptographic Access Provider 128-bit for iSeries (#5727-AC3).
- DB2 UDB for Linux, UNIX, and Windows v8.1 and later

## Data Type Mappings

StarSQL v5 added support for DB2 LOB data types. The DB2 types are mapped to ODBC SQL types as shown in [Table 1](#):

**Table 1. Mapping of DB2 Data Types to ODBC SQL Data Types**

DB2 Type	StarSQL v5 ODBC SQL Type
BLOB	SQL_LONGVARBINARY
CLOB	SQL_LONGVARCHAR
DBCLOB	SQL_LONGVARCHAR

As shown in [Table 2](#), StarSQL v5 changes the mapping for DB2 long strings—they are no longer differentiated from short strings:

**Table 2. Mapping of DB2 Strings**

DB2 Type	StarSQL v4.x ODBC SQL Type	StarSQL v5 ODBC SQL Type
VARCHAR	SQL_VARCHAR	SQL_VARCHAR
LONG VARCHAR	SQL_LONGVARCHAR	SQL_VARCHAR
VARGRAPHIC	SQL_VARCHAR	SQL_VARCHAR
LONG VARGRAPHIC	SQL_LONGVARCHAR	SQL_VARCHAR
VARCHAR FOR BIT DATA	SQL_VARBINARY	SQL_VARBINARY
LONG VARCHAR FOR BIT DATA	SQL_LONGVARBINARY	SQL_VARBINARY

These new mappings affect ODBC SQL catalog query results. For example, `SQLColumns` for a DB2 LONG VARCHAR returns ODBC type `SQL_VARCHAR` for a package bound by StarSQL v5. If StarSQL v4.x uses packages bound by StarSQL v5, catalog queries will return the new StarSQL v5 results, but otherwise the driver should run as before.

Parameters for SQL statements are also affected. If an application binds a parameter as SQL type `SQL_LONGVARCHAR`, it is sent as a DB2 CLOB. StarSQL v4.x sends `SQL_LONGVARCHAR` as a DB2 VARCHAR string.

StarSQL v5 has an optional data source setting, `LongStrParams` (see "[LongStrParams](#)" [on page 109](#)), for applications that require backwards compatibility for `SQL_LONGVARCHAR` (or `SQL_LONGVARBINARY`) parameters. This data source setting does not affect the types returned for result set columns, or for types returned by catalog queries.

StarSQL v5 no longer returns type `SQL_FLOAT` from `SQLGetTypeInfo`. StarSQL maps DB2 REAL to `SQL_REAL`, and DB2 DOUBLE to `SQL_DOUBLE`, and only these two floating point types are returned by `SQLGetInfo`. StarSQL still accepts `SQL_FLOAT`, as a synonym for `SQL_DOUBLE`, as a type, for example, for `SQLBindCol`.

For an introduction to LOB support in DB2 for OS/390, see the IBM Redbook "Large Objects with DB2 for z/OS and OS/390" (SG24-6571), available at <http://www.redbooks.ibm.com>.

## Documentation

There are many sources of information that can help you install, configure, and use the StarSQL driver. The following sections describe the information available from StarQuest and provides references to other information that may be particularly useful.

### Quick Path to Using StarSQL

StarQuest provides StarSQL Quick Start Guides that provide step-by-step instructions for quickly installing and using the StarSQL ODBC and JDBC driver on a particular computing platform. The procedures in the Quick Start Guides are appropriate for the most common environments and describe the fastest way to install and configure the software you need to begin using the driver. If you need to customize the StarSQL driver settings or have an environment for which the default values are not appropriate you can refer to the product documentation for details.

All the Quick Start Guides are listed at <http://www.starquest.com/Supportdocs/browseQuickStarts.shtml>.

### StarSQL Product Documentation

The StarSQL for UNIX product documentation consists of the following components:

- this User's Guide
- Release Notes
- Command Manual Pages

#### User's Guide

This User's Guide provides information about installing the StarSQL software, and configuring a client license to use the driver. It also describes how to use the StarSQL driver and the utilities and programs that are included with it. Licensing is managed by StarLicense server software, which includes separate documentation for installing and configuring a StarLicense server.

#### Organization

The 32-bit and 64-bit versions of the StarSQL driver differ mainly in how they are installed and configured. Therefore, the information about installing and configuring the 32-bit and 64-bit driver is provided in separate chapters and appendixes.

---

<b>To install:</b>	<b>Refer to:</b>
StarSQL 32-bit driver	<a href="#">"Installing 32-bit StarSQL for UNIX" on page 23</a> , or <a href="#">"Manually Installing 32-bit StarSQL for UNIX" on page 71</a>
StarSQL 64-bit driver	<a href="#">"Installing 64-bit StarSQL for UNIX" on page 27</a> , or <a href="#">"Manually Installing 64-bit StarSQL for Linux" on page 79</a>
both 32- and 64-bit StarSQL drivers	<a href="#">"Installing 32-bit StarSQL for UNIX" on page 23</a> , and <a href="#">"Installing 64-bit StarSQL for UNIX" on page 27</a>

---

The remaining chapters and appendixes are common to either version of the driver, with any minor differences noted in context.

### Conventions

The StarSQL driver can be installed to a user-defined location, and differs if both the 32-bit and 64-bit versions of StarSQL are installed on the same computer. The User's Guide shows `$STARDIR` to indicate the location where the StarSQL software is installed. Substitute the `$STARDIR` variable with the full pathname to the installation directory.

You also can specify and export an environmental variable that defines the location of the StarSQL software rather than typing the path in place of the `$STARDIR` placeholder. For example, you could specify an environment variable named `$STARDIR` to specify the location to the 32-bit driver and, if you also install the 64-bit version of StarSQL on the same computer, specify a `$STARDIR64` environment variable to provide the path to that version. The `$STARDIR` placeholder in the documentation refers to the location of whichever version of StarSQL you want to use.

### Release Notes

The Release Notes contain important information about using StarSQL for UNIX in specific environments, known limitations, and a history of changes to the driver software. The Release Notes specific to each version of StarSQL for UNIX are located in the respective 32-bit and 64-bit subdirectories.



## Command Manual Pages

StarSQL for UNIX provides commands for viewing packages, testing connections, administering licenses, and troubleshooting. These commands are described in [Appendix C](#) on [page 81](#) of this User's Guide. You also can obtain information about the commands in the form of man pages, which are located in \$STARDIR/man. Change to the \$STARDIR/man directory and then enter the **man** command along with the name of the StarSQL command you want to display information about. The following example would display the man page for the **starping** command.

```
# cd $STARDIR/man
# man starping
```

## Third-Party Information Resources

The following documents, published by IBM, provide information that you may find helpful in setting up DB2 hosts for access by StarSQL users. The document titles and publication numbers were current when this list was prepared, but are subject to change by IBM.

**Table 3. Related IBM Reference Documentation**

IBM Publication Title	IBM Publication Number
IBM DB2 for OS/390 Version 5 Administration Guide	SC26-8957
IBM DB2 for OS/390 Version 5 Installation Guide	GC26-8970
IBM DB2 for OS/390 Version 5 Reference for Remote DRDA Requesters and Servers	SC26-8964

<b>IBM Publication Title</b>	<b>IBM Publication Number</b>
IBM DB2 UDB for OS/390 Version 6 Administration Guide	SC26-9003
IBM DB2 UDB for OS/390 Version 6 Installation Guide	GC26-9008
IBM DB2 UDB for OS/390 Version 6 Reference for Remote DRDA Requesters and Servers	SC26-9012
DB2 for OS/390 and z/OS V7 Administration Guide	SC26-9931
DB2 for OS/390 and z/OS V7 Installation Guide	GC26-9936
DB2 UDB for OS/390 and z/OS V7 Reference for Remote DRDA Requesters and Servers	SC26-9942
DB2 for OS/390 and z/OS V8 Administration Guide	SC18-7413
DB2 for OS/390 and z/OS V8 Installation Guide	GC18-7418
DB2 UDB for OS/390 and z/OS V8 Reference for Remote DRDA Requesters and Servers	SC18-7424
DB2 for OS/390 and z/OS V9 Administration Guide	SC18-9840
DB2 for OS/390 and z/OS V9 Installation Guide	GC18-9846

<b>IBM Publication Title</b>	<b>IBM Publication Number</b>
DB2 UDB for OS/390 and z/OS V9 Reference for Remote DRDA Requesters and Servers	SC18-9853
DB2 Server for VM v7.3 System Administration	SC09-2980
DB2 Server for VSE v7.3 System Administration	SC09-2981
DB2 Solutions with VSE and VM Implementation and Usage	SG24-2036
IBM DB2 Multisystems for OS/400	SC41-3705, SC41-5705
AS/400 Distributed Database Programming or iSeries Distributed Database Programming	none
IBM DRDA Connectivity Guide	SC26-4783
Distributed Functions of DB2 for z/OS and OS/390	SG24-6952
WOW! DRDA Supports TCP/IP: DB2 Server for OS/390 and DB2 Universal Database	SG24-2212

## Contacting StarQuest

Please use the following methods to contact StarQuest Ventures if you need to obtain a license key, or have suggestions or need information about StarQuest products.

### Support

The easiest method for licensing StarQuest products is to use the online licensing feature of the License Configuration utility (see "[Licensing StarQuest Products](#)" on page 30). If you cannot request a license over the Internet you can obtain a license key for a StarQuest product by sending an email to [contact@starquest.com](mailto:contact@starquest.com) with the following information:

- TCP/IP address or Host ID of the computer on which the license will be installed
- Number of connections purchased
- Company Name
- Contact Name
- Phone Number
- Email Address

StarQuest Support will send a reply email that provides the license key for your organization's use of the product. Since the license is unique to the computer on which it will be installed, you must contact StarQuest should you need to move the license from one computer to another.

Additional technical support may be available subject to the prices, terms, and conditions specified in your organization's maintenance contract with StarQuest Ventures, Inc.

## Sales and Service

If you have ideas for product enhancements or need more information about StarQuest products, please contact us via any of the following methods.

---

Address	StarQuest Ventures, Inc. P.O. Box 1076 Point Reyes Station, CA 94956
Telephone	415-669-9619
Fax	415-669-9639
Email	<a href="mailto:contact@starquest.com">contact@starquest.com</a>
World Wide Web	<a href="http://www.starquest.com">www.starquest.com</a>

---



# Installing 32-bit StarSQL for UNIX

---

This chapter describes how to install and configure the 32-bit version of StarSQL for UNIX. It also discusses issues to consider if you are upgrading from a prior version of StarSQL. Be sure to review the Release Notes included in the distribution for important information about installing or upgrading the StarSQL for UNIX driver.

If you do not already have one or more StarLicense servers set up, you may want to obtain license keys and install and configure the StarLicense software, before you install StarSQL. Licensing for StarSQL for UNIX is further discussed in ["Licensing StarQuest Products" on page 31](#), and the StarLicense server software includes documentation for installing and configuring a license server.

## Client Computer Requirements

Refer to ["The StarSQL for UNIX Driver" on page 9](#) for a description of the network and host requirements for using StarSQL. The 32-bit version of StarSQL for UNIX has been tested to work with the following operating systems. Other distributions of Linux or UNIX may also work with the StarSQL driver.

- IBM AIX v4.3.3 and later with
  - AIX C Set ++ for AIX Application Runtime package (xlC.rte package on the AIX 4.x distribution CD-ROM)
- Fedora 10 and later
- FreeBSD v7.0 and later (x86)
- HP-UX 11.11 and later (PA-RISC)
- Red Hat Enterprise Linux 5.3 and later (x86)

- Sun Solaris 8 and later (SPARC) with
  - the UTF-8 locale ICONV patch:
  - Solaris 8 Patch-ID# 113261-01
- SUSE Linux Enterprise Desktop or Server 10 SP2 and later (x86)
- Ubuntu Desktop or Server Edition v9.04 or later

Use SMIT or lslpp to determine whether the C++ Application Runtime is installed on an IBM AIX computer.

```
lslpp -L | grep xlc.rte
```

If the C++ runtime is not already installed, use **smit install** to install the AIX packages from the AIX distribution CD-ROM.

You also can run StarSQL on HP-UX Itanium if the calling application is a 32-bit PA-RISC binary, on a 64 bit Linux (x64) system if the calling application is a 32-bit x86 binary, or on a 64-bit FreeBSD (x64) computer if the calling application is a 32-bit x86 binary and you install the FreeBSD 32-bit runtime compatibility libraries.

## Upgrading StarSQL

Versions of the StarSQL software are categorized as major releases, point releases, and hot-fixes. Major releases usually are designated with a new major release number such as v4.1, v5.3, or v5.5. Point releases provide defect corrections to a major release, and are designated with a version number such as v5.21, v5.34, 5.51 and so on. If a particular problem arises between scheduled releases, one or more of the StarSQL components may be provided as a “hot-fix” until the fix is incorporated into a point release or a major release.

If you are running a previous version of StarSQL for UNIX, the v5.5 installer will automatically uninstall the prior version before it installs the new version of StarSQL. Before you start the installation of v5.5, review the Release Notes that are included with the StarSQL for UNIX distribution, as they contain important information about upgrading from a prior release.

If you have a customized `.swodbc.ini` file, make a backup copy of it before you install v5.5 of StarSQL as the file will be replaced during the installation. You must use the new version of the `.swodbc.ini` file to take advantage of the support for additional DB2 hosts, such as the v9.1 release of DB2 for Linux, UNIX, and Windows. Copy any custom settings from the backup copy of the `.swodbc.ini` file to the new version of the file after you install v5.5 of the StarSQL driver.

Typically users access the system-wide `swodbc.ini` file on a shared network drive. Having a writeable `.swodbc.ini` file in the home directory allows a user to modify the behavior of global data source settings (see ["Global DSN Keywords" on page 94](#))



and run the **trcstart** command to collect DRDA traces (see ["trcstart" on page 91](#)). If the user home directory contains an `.swodbc.ini` file, the settings in that file take precedence over any settings in the `swodbc.ini` file in `$STARDIR/etc`.

## StarSQL for UNIX 32-bit Driver Installation

You can use the Easy Install method to install StarSQL on all UNIX platforms, or you can manually install the software. Manually installing the software lets you specify where the software is installed, but also requires that you specify environment variables and run a post-installation script. The following steps use the Easy Install method. Refer to ["Manually Installing 32-bit StarSQL for UNIX" on page 71](#) if you prefer to customize the installation.

The StarSQL Easy Install is identical on all supported UNIX platforms. After you have downloaded StarSQL, follow the steps in this section to install the software.

1. Logon to UNIX as `root` user.
2. Change to the directory that contains the StarSQL installer for your version of UNIX.
3. Enter the following command to execute the shell script "setup," which also runs the post-installation script to create the necessary symbolic links and install conversion tables.

```
# ./setup
```

The setup script installs StarSQL to the default location, which is:

```
/usr/lpp/starsql for AIX  
/usr/local/share/starsql for FreeBSD  
/usr/share/starsql on Linux  
/opt/starsql for HP-UX or Solaris
```

The installation script also either installs the unixODBC driver manager that is included in the StarSQL distribution and creates the `odbcinst.ini` file if it does not find an existing driver manager, or adds a [StarSQL] section to an existing `odbcinst.ini` file to define the driver to the active version of the driver manager.

After you perform the Easy Install procedures you need to configure a connection to a StarLicense server and configure StarSQL data sources for applications to use, as described in ["Using StarSQL for UNIX" on page 31](#).

## Removing the StarSQL 32-bit Driver

This section describes how to uninstall the StarSQL for UNIX software from a computer. The procedures differ depending on which version of UNIX you are running. The following table provides guidelines—refer to the documentation for the UNIX operating system you are using for details.

UNIX Variant	Command for Un-installing Software
AIX	<code>installp -u starsql</code>
FreeBSD	<code>rm -rf /usr/local/share/starsql</code>
HP_UX	<code>swremove starsql</code>
Linux	using RPM: <code>rpm -e starsql</code> using tar: <code>rm -rf /usr/share/starsql</code>
Solaris	<code>pkgrm starsql</code>

# Installing 64-bit StarSQL for UNIX

---

This chapter describes how to install and configure StarSQL for UNIX on a 64-bit computer. It also discusses issues to consider if you want to install both the 32-bit and 64-bit versions of StarSQL on a 64-bit computer. Be sure to review the Release Notes included in the distribution for important information about installing or upgrading the StarSQL for UNIX driver.

StarSQL requires a license key to successfully connect to a DB2 host. You may want to obtain license keys, and install and configure the StarLicense software, before you install StarSQL. Licensing is further discussed in "[Licensing StarQuest Products](#)" on [page 31](#).

## Client Computer Requirements

StarSQL for UNIX has been tested with the following Linux operating systems: Other Linux distributions may also work with StarSQL.

- Fedora 10 and later
- Red Hat Enterprise Linux 5.3 and later
- SUSE Linux Enterprise Desktop or Server 10 SP2 and later
- Ubuntu Desktop or Server Edition v9.04 or later

---

### Note

You must use release 2.2.14 or later of the 64-bit unixODBC driver manager on a 64-bit computer. The version of unixODBC that is included with StarSQL meets the minimum requirements and has been tested to work with the 64-bit version of StarSQL.

---

## Determining Application Architecture

Because 32-bit and 64-bit applications can coexist on a 64-bit system, it can sometimes be difficult to determine whether an application is 64-bit or 32-bit. It's important to know the application architecture because 32-bit applications require a 32-bit DSN and ODBC driver manager and 64-bit applications require a 64-bit DSN and ODBC driver manager.

On UNIX systems, you can use the `file` command to tell whether an application is 32-bit or 64-bit. To do this, locate the application binary and then run **file** `<application_binary>`. If the file is 64-bit, the command output will contain "ELF 64-bit" (or something similar such as "ELF-Class64" or "ELF-64"). If the file is 32-bit, the command output will contain "ELF 32-bit".

## Upgrading StarSQL

If there is a prior version of the 32-bit StarSQL driver already installed on the 64-bit computer, remove it before you install the StarSQL v5.5 software. You can remove the existing StarSQL driver from a Linux computer using either the RPM or tar command, as shown below.

Using RPM:

```
rpm -e starsql
```

Using tar:

```
rm -rf /usr/share/starsql
```

After you remove the prior version of the StarSQL driver you can install either or both the 32-bit and 64-bit versions of the v5.5 StarSQL drivers. Refer to ["Installing 32-bit StarSQL for UNIX" on page 23](#) if you want to install the 32-bit version of StarSQL v5.5. Follow the instructions in this chapter to install the 64-bit version.

## StarSQL for UNIX 64-bit Driver Installation

You can use the Easy Install method to install StarSQL, or you can manually install the driver software. Manually installing the software gives you more control over where the software is installed, although the procedures differ depending on which UNIX operating system you are using and are more involved than using the Easy Install. The following steps use the Easy Install method. Refer to ["Manually Installing 64-bit StarSQL for Linux" on page 79](#) if you prefer to use the platform-specific installation instructions.

The StarSQL Easy Install is identical on all supported platforms. After you have downloaded StarSQL, follow the steps in this section to install the software.

1. Logon to UNIX as `root` user.
2. Change to the directory that contains the StarSQL installer for your version of UNIX.
3. Enter the following command to execute the shell script “setup,” which also runs the post-installation script to create the necessary symbolic links and install conversion tables.

```
# ./setup
```

The setup script installs StarSQL to the default location, which for Linux is `/usr/share/starsql64`, and either creates the `odbcinst.ini` file, or adds a [StarSQL64] section to an existing `odbcinst.ini` file to define the driver to the active unixODBC driver manager.

## **Removing StarSQL from a 64-bit Computer**

If you need to remove the StarSQL for UNIX software you can use either the RPM Package Manager or the UNIX remove command, as shown in the following examples.

To remove StarSQL using the RPM Package Manager, issue the following command:

```
rpm -e starsql64
```

The following command forcibly removes the contents of the StarSQL installation directory and its contents without prompting for confirmation:

```
rm -rf /usr/share/starsql64
```



# Using StarSQL for UNIX

---

StarSQL requires a license to successfully connect to a DB2 host. This chapter describes how to license and use StarSQL for UNIX.

## Licensing StarQuest Products

All StarQuest products are licensed for use. Each product setup contains a client module used to configure the specific license option used to enforce the use of the product. The licensing options allow you to use a node-locked license or a floating license.

### Node-Locked License

A **node-locked license** allows you to use the Product on a single computer: Node-locked licenses are only available for computers using Microsoft Windows Operating Systems. With a node-locked license:

- The computer is identified by a unique Host ID.
- The product can run only on the identified computer.
- The product usage may not exceed the limits allowed by the license.

In addition to the setup of the StarQuest product you will be provided with a unique registration code that should be used for the activation of the software license. It is also possible to use the client module to display the HOSTID to request a software license via email or telephone.

The software license should be activated online using the supplied registration code, or manually after communicating a HOSTID with StarQuest and receiving an email response containing a license string.

## Floating License

A **floating license** allows multiple computers using a StarQuest product to share use of the software license. The software license can be used on any computer within a network provided that the number of concurrent requests does not exceed the limit allowed by the license. All StarQuest products for UNIX and Mac OS X *must* use a floating license. StarQuest products for Windows *may* use a floating license. Generally, only one license server need be installed on a network, to service any number of clients.

For a floating license, in addition to the setup of StarQuest product you will be provided with a StarLicense Server setup and a unique registration code to activate the license server. The *StarLicense Server for UNIX User's Guide* contains details about installing and configuring the server software, but in general:

- The StarLicense Server software should be installed on a network server.
- The network server is usually identified by a unique, static IP address.
- The StarLicense Server should be activated online or via e-mail.
- The StarLicense Server controls the total number of concurrent connections within the network.

For a client to obtain a license from a StarLicense server the parameters of the network server where the StarLicense Server is installed are specified in the appropriate client license module on any computer using the StarQuest product.

## Configuring a StarSQL License

StarSQL for UNIX requires that a floating license be used, although the StarLicense Server software can be located on a remote host or co-located on the same computer as the StarSQL driver. To configure StarSQL for UNIX to check out a license from a StarLicense server, you will need to specify:

- the hostname or IP address of the StarLicense Server
- the port on which the StarLicense Server is listening for license requests
- the product ID of the license installed on the StarLicense Server

The hostname, port, and product ID specified for the client must match how that server is configured—contact the administrator responsible for configuring the StarLicense server(s) if you need details about how the server is configured.

For UNIX-based computers the necessary StarLicense client software is installed when you install the StarSQL driver. Licenses configured with either the 32-bit or 64-bit version of StarLicense are available to both 32-bit and 64-bit applications that use StarSQL to access the host.

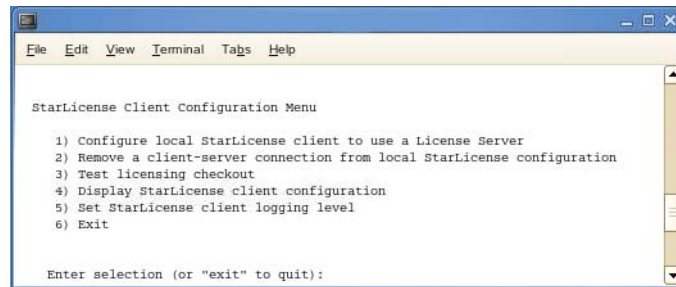


A StarLicense Client Configuration utility allows you to easily specify which StarLicense server to use when the UNIX client computer needs a connection.

1. Log into the computer as `root` user.
2. Change to the directory where StarSQL is installed.
3. Enter the following command to start the configuration utility.

```
# ./config-lic
```

**Figure 2. StarLicense Client Configuration Menu**



4. From the StarLicense Client Configuration Menu, select option 1, “Configure local StarLicense client to use a License Server.”
5. Enter the hostname or IP address of the StarLicense server. (If the StarLicense server software is installed on the local computer you can specify the loopback address 127.0.0.1 or “localhost”.)
6. Enter the number of the port on which the StarLicense server is listening for license requests.
7. Enter the product ID of the license that is configured on the StarLicense server.
8. After the connection to the StarLicense server has been added, select Option 3, “Test licensing checkout,” to verify that a license can be checked out.

Refer to ["StarLicense Client Configuration Utility" on page 117](#) for information about all the options available from the StarLicense Client Configuration Menu.

## Customizing the unixODBC Driver Manager Configuration

Most distributions of Linux include a version of the unixODBC driver manager. The StarSQL driver has been tested to work with the unixODBC driver manager that is included with StarSQL. You can use either the unixODBC binaries that are supplied with your operating system, or the unixODBC included with StarSQL, but do not use both.

---

### Note

The version of unixODBC that is included with StarSQL meets the minimum requirements and has been tested to work with the StarSQL driver. The StarSQL driver requires the following versions of the unixODBC driver manager:

- Release 2.2.12 or later of the 32-bit unixODBC driver manager to use the 32-bit version of the StarSQL driver.
  - Release 2.2.14 or later of the 64-bit unixODBC driver manager on a 64-bit computer.
- 

To determine which version of the unixODBC driver manager is in use, issue the `ldd` command to show the shared libraries that your ODBC application requires access to. The example output below shows that the `isql` program is using the driver manager (`libodbc.so.1`) from the `$STARDIR` directory. The unixODBC driver manager that is included with operating system distributions is typically located in `/usr/lib`. To control which version of unixODBC is used you can either set the `LD_LIBRARY_PATH` environment variable or edit `/etc/ld.so.conf` and run `ldconfig` to update the configuration.

```
$ ldd /usr/share/starsql/odbc/bin/isql
libodbc.so.1 => /usr/share/starsql/odbc/lib/libodbc.so.1
(0x40014000)
libpthread.so.0 => /lib/libpthread.so.0 (0x40081000)
libc.so.6 => /lib/libc.so.6 (0x400d2000)
libdl.so.2=> /lib/libdl.so.2 (0x401f9000)
/lib/ld-linux.so.2 +> /lib/ld-linus.so.2 (0x40000000).
```

The unixODBC driver manager uses the file `odbcinst.ini` to keep track of which ODBC drivers are installed. The location of this file can vary. You can locate the `odbcinst.ini` file using the `odbcinst` program provided with the unixODBC driver manager, as shown below.

```
$ odbcinst -j
```

The `odbcinst` program shows the version of unixODBC that is installed, and the location of the files that specify the ODBC drivers (`odbcinst.ini`), the system DSNs (`odbc.ini`), and the user DSNs (`.odbc.ini`).

The unixODBC driver manager that is supplied with StarSQL looks for the global configuration files— `odbcinst.ini` and `odbc.ini`— in `/usr/local/etc`. If these files are located in a different directory you can define and export the `ODBCSYSINI` environment variable to specify a different location on the local computer. Note that both of the configuration files must be in the location specified by the `ODBCSYSINI` variable. Copy the configuration files from the `/usr/local/etc` directory as these versions of the files contain the DSN and StarSQL driver information.

The default unixODBC driver manager configuration works with most environments. If you need to modify the behavior, such as to turn on ODBC tracing to troubleshoot a problem, use a text editor to modify the `odbcinst.ini` file. The below listing shows sample contents of an `odbcinst.ini` file after installing both the 32-bit and 64-bit versions of the StarSQL driver.

### Figure 3. Sample `odbcinst.ini` file with StarSQL Driver Defined

```
[ODBC Drivers]
StarSQL =Installed

[ODBC]
Trace = 0
Trace File = /tmp/sql.log
InstallDir = /usr/share/starsql/odbc

[StarSQL]
Driver=/usr/share/starsql/lib/libSWODBC.so
Setup=/usr/share/starsql/odbc/lib/libodbcstarsqlS.so
Description=StarSQL
FileUsage=1

[StarSQL64]
Driver=/usr/share/starsql64/lib/libSWODBC.so
Setup=/usr/share/starsql64/odbc/lib/libodbcstarsqlS.so
Description=StarSQL64
FileUsage=1
```

Refer to the unixODBC Readme file, included in the StarSQL distribution, for more information about using the driver manager.

## Configuring Data Sources

A Data Source Name (DSN) definition controls how the StarSQL driver interacts with a specific data source. Each data source to which you want to connect using StarSQL must be defined in a text file named `odbc.ini` for system DSNs, and `.odbc.ini` for user DSNs. If there is a user DSN and system DSN with the same name, the user DSN is used.

Where possible, StarSQL supplies default configuration values in the example `odbc.ini` file that is distributed in the `$STARDIR/etc` directory. However, there are some values, such as specific names used in your environment, which you must configure.

The unixODBC driver manager includes a graphical ODBC Data Source Administrator program, `ODBCConfig`, to make it easy to define the minimum values necessary for a StarSQL DSN. Creating and modifying DSNs using the `ODBCConfig` interface modifies the `odbc.ini` or `.odbc.ini` file, and the `odbcinst.ini` file, as appropriate, with the values you specify using the graphical interface.

Both 32-bit and 64-bit data sources are visible from either version of the ODBC Data Source Administrator, but it is important to use the 32-bit version of the Administrator to create DSNs for 32-bit applications and the 64-bit version of the Administrator to create 64-bit DSNs. You cannot run both the 32-bit and 64-bit versions of the `ODBCConfig` program at the same time.

To run the `ODBCConfig` program you need to log in to the UNIX system from a GUI environment.

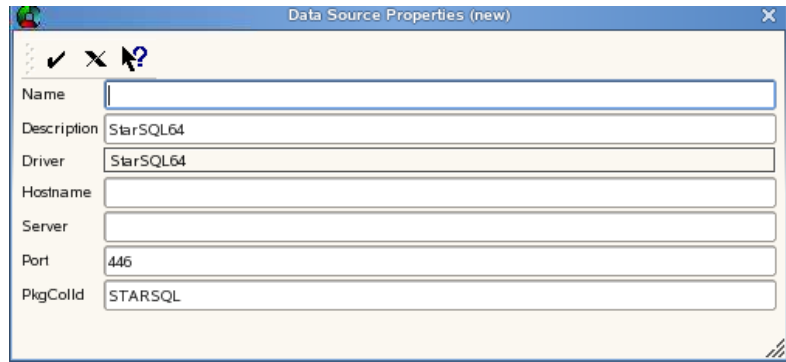
1. If you plan to create system DSNs, log in to the UNIX computer as `root` user. You do not need to be `root` user to create a user DSN.
2. Change to the directory in which the desired version—either the 32-bit or 64-bit—of ODBC Data Source Administrator is installed.
3. Type **ODBCConfig** and press Enter to start the ODBC Data Source Administrator.
4. From the ODBC Data Source Administrator, click on the User DSN or System DSN tab to configure the desired type of data source. The following table describes each type of DSN you can create. Note that, although there is a File tab in the Administrator window, `ODBCConfig` does not currently support

creating a File DSN. You can still use a File DSN with StarSQL if the file is created manually or produced using the SAVEFILE connection attribute to SQLDriverConnect.

<b>DSN Type</b>	<b>Description</b>
User	A User DSN is local to the computer on which it is defined and can be used only by the current user.
System	A System DSN is local to the computer on which it is defined but can be used by the system or any user with privileges for the computer.
File	A File DSN contains information for a single data source that is stored in a file that can be shared by multiple users, or multiple computers if the file is stored on a network file server.

5. To modify an existing data source, select a Data Source Name and click Configure.  
To add a new data source, click the Add button.
6. If you are creating a new data source, a pane entitled Select a Driver appears for you to select which driver you want to use. Select StarSQL or StarSQL64, as appropriate, and click the OK button to display the properties you can configure for the DSN.

7. Specify the connection and host properties that are appropriate for your environment. The following illustration shows example values for a StarSQL DSN.





Provide a unique name for the data source in the Name field. Do not specify a data source name that includes parentheses [ ( ) ]. The ODBC driver manager does not allow parentheses in data source names. If you configure a data source with parentheses in the name, the DSN will not be usable and can be removed only by manually editing the `odbc.ini` or `.odbc.ini` file. System DSN information is written to the `odbc.ini` file in the ODBC system directory (such as `/usr/local/etc` or `$ODBCSYSINI`), and user DSN information is written to the `.odbc.ini` file in the user's home directory.


You must specify a valid host name, and you can optionally specify a different port number or SQL package collection for the data source to use. The data source configuration parameters are further described in the following table.

**Table 4. Required Data Source Configuration Parameters**

Keyword	Description
HostName=	The host name refers to the name of the DB2 host system. Set the HostName field to either a TCP/IP host name (such as host5.mydomain.com) or a static IP address in dotted decimal notation (such as 198.147.235.1).
PkgColID=	The SQL Package Collection ID indicates the location on the DB2 host of the packages required by StarSQL to execute Dynamic SQL.  On DB2 for i, the SQL Package Collection ID is the name of the collection or library that contains the packages. On all other platforms, the Package Collection ID is the name of the virtual collection associated with these packages.  If necessary, obtain the package collection ID from your Database Administrator. The default package collection ID is STARSQL.
Port=	The Port designates the IP port on which the DB2 host is listening for incoming TCP/IP connection requests. The default port for DRDA communications is 446.
Server=	Enter the name of the database server that will be accessed through this data source. You may need to obtain the database server name from the Database Administrator.  The database server name is known by different names depending on the DB2 host implementation. On DB2 for OS/390 it is called the location name, on DB2 for i it is called the relational database name (RDB), and on DB2 UDB for Windows and UNIX, it is the name of the database.

Click the Help icon  and click within a field to display information about the data source property.

Click the Save icon  to save the DSN definition.

Click the Cancel icon  to exit the dialog without saving changes.

Refer to ["Customizing the StarSQL for UNIX Configuration" on page 93](#) for information about configuring global data source settings and a complete list of the DSN settings that are available for customizing the behavior of the StarSQL driver.

## Testing the StarSQL/DB2 Connection

After you install and configure StarSQL for UNIX, perform the procedures in this section to test that the client can connect to the DB2 host using StarSQL. To establish connectivity to a DB2 database, the host must be configured to accept connections from StarSQL. Refer to ["Preparing Hosts for StarSQL Access" on page 45](#) if your host is not already configured for StarSQL connections.

### Testing the Network Connection

You can use the **starping** command to test the network connection between StarSQL and the DB2 host. The **starping** command tests only the ability to connect to the DB2 host—it does not bind packages on the host or test that you have a valid license. Use the **simpleconn** command, described in the next section, to test for connectivity and proper configuration of StarLicense and the ODBC data source.

To run the **starping** command, enter the command and respond to the prompts for the data source name, user name, and password:

```
$ starping
Enter Data Source: data source name
Enter User Name: userid
Enter Password: password
```

A successful connection returns the following message:

```
Connection Succeeded!
```

See [page 90](#) for more information about the **starping** command. If the connection does not succeed, make sure that the host has been properly configured for StarSQL connections, as described in ["Preparing Hosts for StarSQL Access" on page 45](#). Contact Technical Support if you need further assistance.

### Testing the unixODBC Driver Manager Connection

StarSQL for UNIX includes a program named **simpleconn** that allows you to test that StarSQL can connect to a data source using the unixODBC driver manager. You must specify a user name and password of an account that is valid on the host and has permission to access the database. (See ["Preparation Required for All Hosts" on page 45](#) for more information about user accounts and permissions.)



Enter the following command, specifying a Data Source Name (DSN) and user account that is valid in your environment:

```
simpleconn DSN UserId Password
```

For example:

```
$ simpleconn DSN1 staruser starpass
Connection #1
    Driver Name: libswodbc.a
    Driver Version: 5.30.1903
    Database Name: DB2 Universal DataBase
    Database Version: 08.021.0007
Connection #2
    Driver Name: libswodbc.a
    Driver Version: 5.30.1903
    Database Name: DB2 Universal DataBase
    Database Version: 08.021.0007
```

Upon successful connection, the **simpleconn** command returns the database name, platform, and version, and the driver name and version. Refer to [page 86](#) for more information about using the **simpleconn** command.

## Using the isql Utility to Test Connections

The version of unixODBC that is included with StarSQL includes the isql command line utility. You can use isql to test a connection but it is designed to be used by those experienced with the Structured Query Language (SQL). The isql utility allows you to:

1. connect to a Data Source (using a DSN)
2. send SQL commands to the Data Source
3. receive results from the Data Source

To determine which copy of isql is active, issue the following command.

```
$ which isql
/usr/share/starsql/odbc/bin/isql
```

The example output shows that the version of isql in the \$STARDIR directory is active. The version of isql that is included with operating system distributions is typically located in /usr/bin. Set your PATH environment variable to determine which version of isql is used.

In addition to supplying ad-hoc query commands such as `SELECT * FROM MYTABLE`, the "help" command can be used to make catalog calls. Type "help help" to see a list of available commands.

## Using StarSQL with ODBC Applications

After you have configured data sources and tested that the StarSQL driver can use the unixODBC driver manager and check out a license, you can use StarSQL with any ODBC application to connect to a DB2 host. The method for selecting which data source to use from a specific application varies with each application; refer to the documentation for your application if you need details about how to configure the application to use a particular data source.

---

### Note

To use StarSQL and the unixODBC driver manager with the JDBC-ODBC Bridge on AIX, include `$STARDIR/odbc/lib.ar` in the LIBPATH variable in place of `$STARDIR/odbc/lib`. Do not include both `$STARDIR/odbc/lib` and `$STARDIR/odbc/lib.ar` in the LIBPATH variable as only the first one will be used. If you need to use multiple library path definitions, change the LIBPATH variable as needed, or create a shell script that appropriately sets the environment variable before invoking the application that uses the environment.

---

## Developing ODBC Applications for StarSQL

This section describes how to integrate an ODBC application to use the StarSQL driver and unixODBC driver manager.

---

### Note

The FreeBSD version of StarSQL is built with libraries that use FreeBSD userland threads. If you are developing applications on a computer running FreeBSD, use `g++` to build applications that use the StarSQL driver. Refer to the sample build script `$STARDIR/samples/simpconn/build` for an example of building applications in a FreeBSD environment.

---

When developing applications that will be used with the unixODBC driver manager supplied with StarSQL, use the header files in `$STARDIR/odbc/include`, and link with the libraries located in `$STARDIR/odbc/lib`. Include the library directive `-lodbc` to include `libodbc.a` for AIX and `libodbc.so` for FreeBSD, Linux, or Solaris, and `libodbc.si` for HP-UX.

Following is an example for FreeBSD, Linux, and Solaris:

```
ODBCHOME=${STARSQL}/odbc;export ODBCHOME
cc program.c -I${ODBCHOME}/include \
-L${ODBCHOME}/lib -lodbc -lc -o $program
```

See the shell script "build" in \$STARDIR/samples/simpconn for examples of building a simple ODBC application that uses StarSQL to connect to a host.



# Preparing Hosts for StarSQL Access

---

This chapter describes how to prepare host systems to enable StarSQL to provide access to the host databases.

It covers:

- Preparation required for all hosts
- Preparing a DB2 for OS/390 host
- Preparing a DB2 for i host
- Preparing a DB2 UDB for Linux, UNIX, and Windows host
- Preparing a DB2 Server for VSE & VM

These sections cover details of the DB2 environment that are pertinent to StarSQL. For complete documentation of installation and configuration on the host, consult IBM's DB2 documentation, especially IBM's *DRDA Connectivity Guide* (see "[Third-Party Information Resources](#)" on [page 17](#) for details).

Contact StarQuest Customer Support (see [page 20](#)) for assistance if you plan to use StarSQL to connect to DB2 UDB using the SNA network protocol.

## Preparation Required for All Hosts

Regardless of the host platform, you will need the information described in [Table 5](#) to configure an ODBC data source that will use StarSQL to access the DB2 host. You may need to obtain this information from the DB2 administrator. The section "[Sample DSN File](#)" on [page 97](#) contains details about configuring data sources.

**Table 5. Host Information Required for Data Source Configuration**

Data Source Information Item	Information Needed
Package Collection Name	The location of the SQL packages that StarSQL requires. The Package Collection Name on a DB2 for OS/400 host is the name of the library that contains the StarSQL packages. For DB2 for OS/390 and VSE & VM, it is the name of the virtual collection associated with these packages. Set the Package Collection ID to SQLDBA for a DB2 for VSE & VM host.
Database Server Name	The relational database name. On different DB2 hosts this may be referred to as Location Name, Global Resource Name, RDB name, Database Name, or dbname.
User ID and Password	A valid user id and password for logging into the database. The user account information is not stored in the DSN.
TCP/IP Connection Information	The host name and the port used for DRDA communications.

In addition to properly preparing the host, each StarSQL user must have an account on the host and have permission to access the necessary packages.

## User Accounts

To connect to a DB2 database, each StarSQL user needs an account on the host database. An account consists of a user ID and password.



You need to provide the user account information to each StarSQL user who needs to connect to the database.

## Permissions

Usually a database administrator (DBA) is responsible for packages on the host, including binding them and granting permissions to use them. Depending on the host platform and the type of package used by the ODBC application, the DBA may need to grant StarSQL users explicit permissions to access data used by the application.

## Preparing DB2 on an OS/390 Host

Preparing DB2 on an OS/390 host for access with StarSQL primarily involves configuring the Distributed Data Facility (DDF), which is a component of DB2 for OS/390. Its primary task is to process DRDA requests. DDF must be active for a desktop to connect to DB2 using StarSQL or any other DRDA requestor or client.

## Configuring DDF

If your organization has not implemented distributed database capabilities, DDF may not be configured and activated. The DSNTINST CLIST provides two panels—DSNTIPR and DSNTIP5 for customizing a DB2 for OS/390 subsystem to use native DRDA TCP/IP support. The DSNTIP5 panel is specific for TCP/IP. However, to use native TCP/IP support, you also must have APPC support configured and active because DB2 uses the network ID and the LU name to identify units of work. You specify the LU name that identifies the DB2 subsystem to VTAM and to uniquely identify logical units of work, in the DSNTIPR panel.

The values specified on these panels are used to generate the JCL that stores them in the DB2 bootstrap data set (BSDS) communication record.

If you are installing DB2, use the DDF panel DSNTIPR and DSNTIP5 to provide the following parameters. To change the DDF parameters after installation, run a customized configuration job DSNTIJUZ to update the BSDS.

- DDF Location Name. This name must be specified for the Database Server Name of the ODBC data source that StarSQL uses to connect to the host.
- Password used when connecting DB2 to VTAM, if a password is required.
- IP port to use for TCP/IP access. To enable support for TCP/IP, set the DRDA port in the DDF to 446.

- IP port to use for two-phase commit. The RESYNC PORT parameter in the DSNTIP5 panel specifies a TCP/IP port number for processing requests for two-phase commit re synchronization. The RESYNC PORT must be different than the DRDA PORT, and it must match the port number specified for two-phase-commit recovery operations in the StarSQL Resource Manager. The StarSQL Resource Manager uses a default port value of 5020.

For more information about establishing connectivity between your desktop and DB2 for OS/390 with TCP/IP, consult the Installation Guide for your version of DB2 for OS/390 (see "[Documentation](#)" on page 15 for document details).

For more information about configuring DDF, consult IBM's DB2 for OS/390 installation documentation and the IBM Redbook, *Distributed Functions of DB2 for z/OS and OS/390*. For more information about establishing connectivity between client computers and DB2 for OS/390 over a TCP/IP network, the IBM Redbook, *WOW! DRDA Supports TCP/IP: DB2 Server for OS/390 and DB2 Universal Database*, may be particularly useful. Refer to "[Documentation](#)" on page 15 for details about these publications.

## Starting DDF

Use the following command, which requires authority of SYSOPR or higher, to start DDF:

```
-START DDF
```

When DDF starts successfully, the following messages are displayed:

```
DSNL003I - DDF IS STARTING
DSNL004I - DDF START COMPLETE LOCATION locname LU
netname.luname
```

If DDF has not been properly installed, the START DDF command fails and displays the following message:

```
DSN9032I - REQUESTED FUNCTION IS NOT AVAILABLE
```

If DDF has already been started, the START DDF command fails and displays the following message:

```
DSNL001I - DDF IS ALREADY STARTED
```

The following command shows whether DDF is running and, if so, the parameters that it is using:

```
-DIS DDF
```



## Supporting Password Management Using DRDA Flows

Password Management using DRDA flows is supported for a network using the TCP/IP protocol. There are two host requirements to support the ability of StarSQL users to change their host passwords through StarSQL on DB2 for OS/390 v5 and later.

If you are using DB2 for OS/390 v5, install the maintenance fix PTF UQ21052. The IBM APAR PQ15977 describes the problems fixed by this PTF. This maintenance fix is not required for later releases. You also need to set Extended Security to YES (EXTSEC=YES). The default is NO. This can be done using either

- the DSNTIPR (DDF) panel on the DB2 installation dialog.
- a customized configuration job DSNTIJUZ, with the option EXTSEC=YES specified.

## Using StarSQL with Stored Procedures

Stored procedures are application programs that reside on the host and are invoked via DB2. They are usually written in a traditional programming language like COBOL, RPG, or C. They may contain SQL statements for accessing the DB2 database or they may be used to access non-DB2 resources.

You invoke a stored procedure using the SQL Call statement and receive output data in a result set or in output parameters. The Call statement is executed as any other SQL, using SQLExecute or SQLExecDirect.

## Registering Stored Procedures

The stored procedure should be registered on the host so that calling application can obtain information using the SQLProcedures and SQLProcedureColumns functions. The mechanism of registering a stored procedure varies according to the host platform and version of DB2 being used.

For a host running DB2 for OS/390 v5.1, set up a DB2 stored procedure address space and modify the table SYSIBM.SYSPROCEDURES. The parameters specified in SYSIBM.SYSPROCEDURES.PARMLIST must include Procedure Name and IN/OUT/INOUT specifications, as shown in the following example.

```
//MYJOB JOB (ACCOUNT) , 'DB2' ,MSGLEVEL=(1,1) ,
// USER=SYSADM , PASSWORD=MYPASS ,
// CLASS=A , TIME=1440 ,MSGCLASS=X ,NOTIFY=MYUID
// *
// * INSERT A RECORD INTO SYSIBM.SYSPROCEDURES
// *
//TSO EXEC PGM=IKJEFT01 ,DYNAMNBR=60 ,PARM=' PROFILE NOPREFIX '
//STEPLIB DD DISP=SHR ,DSN=DSN510.RUNLIB.LOAD
```

## Preparing Hosts for StarSQL Access

```
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
    DSN SYSTEM(DSN1)
    RUN PROGRAM(DSNTIAD) PLAN(DSNTIA51)
    END
/*
//SYSIN DD *
INSERT INTO SYSIBM.SYSPROCEDURES
( PROCEDURE, AUTHID, LOADMOD, COLLID, LANGUAGE, IBMREQD,
RUNOPTS, PARMLIST, PGM_TYPE, EXTERNAL_SECURITY,
COMMIT_ON_RETURN )
VALUES
( 'MYPROC', ' ', 'MYPROC', ' ', 'C', 'N', ' ',
'P1 VARCHAR(255) INOUT,
P2 VARCHAR(99) IN,
P3 VARCHAR(99) IN,
P4 VARCHAR(99) IN',
'M', 'N', 'Y' );
//
```

For a host running DB2 for OS/390 v6.1 and later, use the CREATE PROCEDURE command to register the stored procedure in the system. The CREATE PROCEDURE command automatically updates the SYSIBM.SYSRoutines catalog table.

```
CREATE PROCEDURE SYSPROC.STARPING (
    IN REGION CHAR(8) CCSID EBCDIC,
    IN PROGRAM CHAR(8) CCSID EBCDIC,
    IN TRANSID CHAR(4) CCSID EBCDIC,
    IN COMLEN SMALLINT,
    INOUT COMMAREA VARCHAR(32700) FOR BIT DATA,
    OUT RC INTEGER,
    OUT ABCODE CHAR(4) CCSID EBCDIC
)
PARAMETER STYLE GENERAL
LANGUAGE C
EXTERNAL NAME 'STARPING'
RESULT SETS 0
DETERMINISTIC
NO SQL
NO DBINFO
NO COLLID
ASUTIME NO LIMIT
NO WLM ENVIRONMENT
```

```
STAY RESIDENT NO
PROGRAM TYPE MAIN
SECURITY DB2
COMMIT ON RETURN YES
```

## Calling Stored Procedures

If you are calling a stored procedure on DB2 for OS/390 v5 and later, you can get a result set from the stored procedure call.

Each of the following SQL statements for calling the sample stored procedure is valid:

```
Call MyProc (1, 'A', ?, ?)
Call MyProc( parm1=1, parm2='A', parm3=?, parm4=?)
Call MyProc( parm1=1, parm2='A', ?, ?)
Call MyProc( ?, ?, ?, ?)
Call MyProc( 1, 'A', parm3=?, parm4=?)
```

## Preparing a DB2 for i Host

This section discusses setting up a DB2 for i host for supporting a connection through StarSQL for Windows.

It covers:

- creating a library/collection for SQL packages
- determining the RDB name
- enabling DRDA over TCP/IP for OS/400
- installing stored procedure for password management
- supporting password encryption, LOB data types, and two-phase commit transactions

For complete information about setting up DB2 for i, consult IBM's DB2 for i installation documentation (see "[Documentation](#)" on page 15 for detailed references).

## Creating a Library for SQL Packages

On DB2 for i, required SQL packages are stored in a collection, or library. You may need to create the collection or library on the host.

Use the CRTLIB command to create a new library for the SQL packages used by StarSQL. You can do this from a 5250 terminal session with a user ID that has QSECOFR privileges. The library does not have to be a SQL collection, but it must be accessible to all StarSQL users.

For example, the following command creates a new library named STARSQL:

```
CRTLIB STARSQL
```



Record the name of the library as it must be specified as the SQL Package Collection ID in the data source configuration.

---

### Determining the RDB Name

Determine the Relational Database (RDB) name of the DB2 for i. From the DB2 for i command line, enter:

```
WRKRDBDIRE
```

Look for an entry with a Remote Location value of \*LOCAL. If such an entry does not exist, create it with the I=ADD option. A common convention is to use the same name as the DB2 for i system name for the RDB name.



Make a note of the RDB name as you need to specify it as the Server in the data source configuration.

---

### Enabling DRDA Over TCP/IP

The Distributed Data Management (DDM) server allows client computers to access the DB2 functions. The DDM server supports remote SQL access, record level access, and remote journals. To initiate a DDM server job using TCP/IP communications a DRDA application or DDM source system connects to the well-known port number for TCP/IP, port 446 or 447. The DDM listener program, upon accepting the connection request, issues an internal request to attach the client's connection to a DDM server job.

The DDM listener program runs in a batch job in the QSYSWRK subsystem. There is one listener program that serves potentially many DDM server jobs. If you have access to iSeries Navigator you can verify whether DDM is configured by selecting TCP/IP from the Network->Servers menu.

Follow the steps below if you need to configure a host running DB2 for i v4r2 and later to accept DRDA requests over TCP/IP:

1. Log on to the DB2 for i host.
2. Change the DDM TCP/IP Attributes to automatically start the listener program by entering:

```
CHGDDMTCPA AUTOSTART (*YES)
```

3. Start the TCPIP DDM Server by entering:

```
STRTCPSVR SERVER (*DDM)
```

When you are logged on to the DB2 for i host, you can examine which port DB2 for i is using to listen for DRDA requests using either the WRKSRVTBLE or the WRKTCPSTS command.

To use the WRKSRVTBLE command:

1. Enter WRKSRVTBLE.
2. Look for the DRDA entry with the port number.

To use the WRKTCPSTS command:

1. Enter WRKTCPSTS.
2. Choose option 3, "Work with TCP/IP connection status."
3. Find the entry with port "drda" and press "F14=Display port numbers." The default port number for DRDA is 446.

## Using StarSQL Stored Procedure for Password Management

StarSQL users can change their passwords on most DB2 hosts using DRDA security flows over TCP/IP. However, that functionality is not available on OS/400. To allow StarSQL users to change their passwords on DB2 for i hosts you must install and register a StarSQL stored procedure on a host that is running DB2 v3r2 and later.

The stored procedure is DRDA11DE, and can be installed with the AS/400 application that is available from the EXTRAS\AS400 directory of the StarSQL installation image. Refer to the Readme file in this directory for details about installing and registering the stored procedure.

The StarSQL Help describes how to configure the data source to use the stored procedure for password management, and how users can change their password from StarSQL.

## Registering Stored Procedures on AS/400

This section describes issues regarding stored procedures that are specific to AS/400 hosts. Refer to ["Using StarSQL with Stored Procedures" on page 49](#) for general information about using stored procedures.

Following is sample SQL for registering a stored procedure on an AS/400 system. It assumes that the COBOL program MYLIB.MYPRGM already exists on the AS/400. This statement modifies the QSYS2.SYSPROCS and QSYS2.SYSPARMS catalog tables for you.

```
CREATE PROCEDURE MYLIB.MYPROC (INOUT PARM1 CHAR(10))  
EXTERNAL NAME MYLIB.MYPRGM LANGUAGE COBOL GENERAL
```

In the above example, the procedure name is MYLIB.MYPROC, which references the COBOL program MYLIB.MYPRGM. The program takes one input parameter called PARM1 which is a char field of length 10. This procedure does not return a result set.

Refer to the *IBM SQL Reference and SQL Programming Guide* for the appropriate version of AS/400 for more information on registering a stored procedure and the full syntax of the CREATE PROCEDURE statement.

## Considerations for Specific AS/400 Releases

In general, it is a good idea to stay current on PTF packages and the DB2 Group PTF, and to use the latest version of StarSQL. The following sections describe more specific issues for particular versions of AS/400.

### OS/400 V6R1 and Later Issues

If you are using V6R1, ensure that the following PTFs are either installed or superseded:

- cum C8064610 or later
- PTF 5761SS1-SI30581

### OS/400 V5 Issues

In addition to the general PTFs that may need to be applied to your OS/400 host, this section describes considerations for supporting password encryption, two-phase commit transactions, and LOB data types on a DB2 UDB host that is running V5R1, V5R2, V5R3, or V5R4.

### **V5R4 Considerations**

If you are running V5R4 of the i5/OS, ensure that the following PTFs are either installed or superseded:

- PTF 5722SS1-SI23461
- PTF 5722SS1-SI24317

If you are using StarSQL to invoke Java stored procedures on an AS/400, make sure the following PTF is installed:

- PTF 5722SS1-SI22551

If your host needs to support BLOB data, make sure the following PTFs are installed:

- PTF 5722SS1-SI22324
- PTF 5722SS1-SI22335

### **V5R3 Considerations**

If you are using V5R3, you may encounter a problem with incorrect precision information being returned for decimal and numeric fields after installing CUM PTF C5298530 or DB2 Group PTF SF99503-8 (November 2005). Install the following PTF to resolve the problem:

- PTF SI121373 for APAR SE23286

If you are using StarSQL to invoke Java stored procedures, install the following PTF:

- PTF 5722SS1-SI18139 for APAR SE20127

### **V5R2 Considerations**

If you are running V5R2, ensure that the following PTF is either installed or superseded:

- PTF 5722SS1-SI18984 for APAR SE21110

If you are using StarSQL to invoke Java stored procedures, install the following PTF:

- PTF 5722SS1-SI18984 for APAR SE21110

After installing the cum PTF or DB2 Group PTF from December 2005, you may experience extremely high temporary storage usage when an ODBC application (such as SQDR) uses StarSQL to connect to the iSeries host. If you encounter this problem, install the following PTF:

- PTF 5722SS1-SI23370 for APAR SE25005

## V5R1 Considerations

To support password encryption on a DB2 UDB for iSeries host that is running V5R1 you must install the Cryptographic Access Provider 56-bit for iSeries (#5722-AC2) or Cryptographic Access Provider 128-bit for iSeries (#5727-AC3).

To support two-phase commit transactions, be sure the following patch has been applied to the OS/400 V5R1 host:

- PTF SI09453, which corrects APAR SE10974: Calling a stored procedure over DRDA results in parameters being tagged as char data when the parameters are binary data.

StarSQL is initially configured to provide LOB support for hosts running OS/400 V5R2 and later. To support LOB data types on an OS/400 V5R1 host, you must edit the `swodbc.ini` file and install all LOB-related PTFs.

**Editing the SWODBC.INI File.** The SWODBC.INI file usually is located in the `$STARDIR/odbc/etc` directory on a UNIX-based computer. To support LOB data types, edit the `swodbc.ini` file to change the following line from:

```
QSQ0501=QSQ44NT.DLL
```

to

```
QSQ0501=QSQ52NT.DLL
```

**Installing LOB PTFs.** IBM has issued several important LOB-related PTFs for OS/400 V5R1. PTF SI05363 implements the iSeries SYSIBM ODBC common catalog views for DB2 UDB family consistency, and SI06891 corrects an APAR with the code that installs SYSIBM. Be sure all LOB-related PTFs have been applied to the host. It is recommended that you install the latest DB2 Group PTF SF99501.

## AS/400 V4R4 Issues

The following are known issues with StarSQL and OS/400 V4R4 that are resolved by OS/400 PTF's:

- function check on certain catalog SQL (also a problem with V4R3): install PTF SF59100 (V4R3) or PTF SF59120 (V4R4) for APAR SA84253.
- function check after repeated iterations of certain catalog SQL and SELECT statements: install PTF SF59294 for APAR SA85033.
- invoking a stored procedure returns a “DDM code point 2114 not supported” error message: install PTF SF58942 for APAR SA84890.



- DRDA problem: SF62126 for APAR SA88584
- DRDA problem: SF62392 for APAR SA88896

## **Preparing a DB2 UDB Host**

This section provides details for setting up a DB2 UDB for Linux, UNIX, and Windows host to support a connection through StarSQL for Windows. It covers:

- Enabling DRDA Support for TCP/IP
- Locating the Database Name

For complete information about setting up DB2 UDB, consult IBM's DB2 UDB installation documentation.

### **Enabling DRDA Support for TCP/IP**

When using TCP/IP to connect to a DB2 UDB host, make sure that the host has a static IP address. You may experience problems installing DB2 UDB on a computer using DHCP. To be recognized, UDB requires an entry in the DNS (Domain Name Server) or an entry in the HOSTS file.

Although you can configure StarSQL to communicate with DB2 using any available port, port 446 is the standard port used for DRDA communications and is the default value that StarSQL uses if another is not specified. DB2 UDB uses a default value of 50000 for DRDA communications. You can view and change the port that DB2 uses from either the DB2 Control Center or from a DB2 command window, as described in the following sections.

If there is a firewall that monitors network traffic to the DB2 host, be sure that it allows DRDA communications to pass through the port that you configure for DRDA requests.

### **Using DB2 Control Center to Specify the DRDA Port**

From a Windows or Linux computer you can use the DB2 Control Center to view and change the port number that the DB2 host uses to listen for DRDA requests.

1. In the DB2 Control Center, right-click the DB2 instance, select Setup Communications, and select the TCP/IP option.
2. Select Properties.
3. Verify the port number that is configured for DRDA communications. The standard port number for DRDA communications is 446.

If you change the configuration, you must restart the instance for the changes to take effect.

## Using db2 Commands to Specify the DRDA Port

As an alternative to using the DB2 Control Center, you can issue db2 commands to change the port that DB2 uses for DRDA communications.

1. Enter the following command to determine on which port DB2 is listening to for TCP/IP communications.

```
db2 get dbm configuration
```

In the dbm configuration, look for “TCP/IP Service Name (SVCENAME).” It will have a value similar to “db2c\_DB2.” This is the symbolic name of the connection port.

2. Find the symbolic name of the connection ports in the services file, which is typically located in /etc/services.
3. Edit the services field and change the value of the TCP/IP connection port to 446 and set the interrupt port to 447.
4. Restart the DB2 instance for the changes to take effect.

To verify that DB2 is listening on the correct port, from either the client or the server enter:

```
telnet <host> 446
```

If DB2 is listening on that port, no error is returned to the telnet window. If DB2 is not listening on that port, you will see an error similar to the following and may need to contact your DB2 administrator for the correct port number to use:

```
Could not open connection to the host, on port 446: Connect failed.
```

Click the Close (X) icon to close the telnet window.

## Enabling Encryption

If you configure the StarSQL driver to send encrypted user IDs and passwords (see ["UseEncryption" on page 113](#)), be sure to enable the database for encryption. On a DB2 for Linux, UNIX, and Windows host you enable encryption by setting the Server Connection Authentication (SRVCON\_AUTH) parameter to “Server encrypt” and restarting the instance.

You can set the Server Connection Authentication parameter either from the Control Center or from a db2 command windows, as described below.

## Using the DB2 Control Center to Enable Encryption

From a Windows or Linux computer you can use the DB2 Control Center to enable encryption.

1. From the DB2 Control Center, right-click the desired database instance and select Configure Parameters from the context menu.
2. Under the Administration category of the Database Configuration window, change the value for the Server Connection Authentication parameter to “Server encrypt” and click OK.
3. Click OK to save the database configuration changes, and then restart the database instance for the change to take affect.

## Using db2 Commands to Enable Encryption

As an alternative to using the DB2 Control Center, you can issue db2 commands to enable encryption.

1. To enable encryption from a db2 command window, enter the following command:

```
db2 update dbm cfg using SRVCON_AUTH  
SERVER_ENCRYPT
```

2. Stop and restart the instance by issuing the following commands:

```
db2stop  
db2start
```

## Locating the Database Name

When an administrator sets up a UDB system, they assign names to DB2 databases. You will need to enter the Database Server Name when you configure data sources on the desktop. You can display a list of the databases that have been created by issuing the following command:

```
db2 list database directory
```

Contact your database administrator if you need to determine which databases should be accessible to StarSQL clients.

## Preparing a DB2 Server for VSE & VM

Support for the TCP/IP network protocol was made available beginning with DB2 for VSE v7.1 and DB2 for VM v6.1. Version 6.1 of DB2 for VM also was the first release to support stored procedures. Contact StarSQL Customer Support if you want to use StarSQL over an SNA network to access a DB2 Server for VSE or VM.

When you configure the StarSQL data source, specify the **dbname** parameter as the Database Server Name in the data source. For VSE the database name typically is defined in the **DBNAME** directory to allow the database administrator to ensure that a unique TCP/IP port number is assigned for each DB2 Server for VSE. The application server must be able to determine on which TCP/IP port to listen for connections. The DBNAME directory contains an entry for each DB2 server that specifies the **dbname** and the TCP/IP port number to use (the TCPPORT parameter). The TCP/IP port also can be specified using the TCPPORT initialization parameter, as described in the *DB2 Server for VSE & VM Operation* manual.

For VM the **dbname** parameter is specified in the CMS Communications Directory in the CMS file of type **NAMES**.

Refer to your IBM documentation for details about configuring DB2 Server to support DRDA connections over a TCP/IP network. The *System Administration Guide* for VSE v7.3 is publication number SC09-2981, and for VM v7.3 the publication number is SC09-2980.

# Binding Packages

---

A SQL package is an object that DB2 uses to process a SQL statement. Different packages are required to execute dynamic SQL, static SQL, and ODBC catalog functions.

StarSQL assumes that the dynamic SQL package and the catalog package already exist in the host database. On the initial connection with the host, StarSQL searches for the required packages, and if it does not find them, it creates them automatically.

The names and locations of the packages that are bound vary, depending on how the data source is configured when the initial connection is made.

The StarSQL user requires certain permissions to bind and use packages on the host. Typically, the package will be created by a database administrator and other users will be granted permission to use the packages.

You can use the StarAdmin utility, which is available from StarQuest, to bind packages on a host. Packages that are bound using StarAdmin can be used by any 32-bit or 64-bit version (Java, Linux, UNIX, or Windows) of the StarSQL driver.

## Compatibility Among SQL Catalog Packages

Major releases of StarSQL also may require changes to SQL catalog packages on the host. StarSQL uses the SQL packages to provide added functionality. (Point releases and hot-fixes usually do not involve changes to the SQL packages.) It is recommended that the SQL packages be rebound when a major version of StarSQL is deployed throughout an organization, or when directed by the instructions included in the Release Notes included with all StarSQL installation images.

Significant changes were made to catalog packages and dynamic SQL packages with the v5.1 and v5.3 releases of StarSQL. Users may get different results from ODBC catalog functions when using a version of StarSQL prior to v5.1 with packages bound that are bound with StarSQL v5.1 or later.

[Table 6](#) shows the level of functionality supported with SQL packages created by different versions of StarSQL. In general, packages bound by later versions of StarSQL are backward compatible with earlier versions of StarSQL, unless specifically noted. If an earlier version of StarSQL cannot function properly with packages that are bound with a later release of StarSQL, you can maintain separate package collections or upgrade all clients to use the latest version of the driver.

**Table 6. Functional Package Differences Among StarSQL Versions**

From	To	New Functionality
v4	v5.1	Packages must be rebound to use LOB data types. Upgrade all clients or maintain a separate package collection for clients running a StarSQL release prior to v5.1
v5.1	v5.3	Backwards compatible. Rebind packages to support use of jumbo packages, or to access DB2 for z/OS when the host is configured to use a multi-byte character set. Refer to the descriptions for UseJumboPackages and CustomizePrdid in <a href="#">"Sample DSN File" on page 97</a> for additional information.
v5.3	v5.5	Rebind packages if you upgrade from a release earlier than v5.38. Client computers that use StarSQL v5.1 or earlier should be upgraded to v5.5 or use a different package collection.

StarSQL for UNIX automatically binds host packages upon the initial connection to the host. To explicitly rebind packages, drop the packages on the host and reconnect with the new version of StarSQL for UNIX.

## Binding Packages with the Explain Option Enabled

DB2 provides an explain facility that provides detailed information about the access plan and environment of static or dynamic SQL statements. Beginning with v5.4 the StarSQL driver includes support for a new keyword in the `swodbc.ini` initialization

file that binds all the StarSQL packages with the EXPLAIN bind option set to ALL. Refer to ["Explain" on page 95](#) if you want to enable the EXPLAIN bind option for all packages that are bound by StarSQL.

## Catalog Package

The catalog package is needed to execute ODBC functions that query the system catalogs. The value of the CatQual keyword (see [page 104](#)) in the data source configuration indicates the name of the catalog package.

## Dynamic SQL Packages

There are several packages used for executing dynamic SQL. The default dynamic package varies by host platform; other dynamic packages will be used based on the IsolationLevel, HeldCursors, KeepDynamic, and BindRules keywords in the data source configuration. See ["Customizing the StarSQL for UNIX Configuration" on page 93](#) for more information about these keywords.

[Table 7](#) shows the dynamic packages that can be bound; the default packages vary for different hosts and are shown in **boldface** type.

**Table 7. Dynamic Packages**

Package Name	IsolationLevel and HeldCursors Setting
<b>SWNC0000</b>	No Commit/Held Cursors=No (default on <b>AS/400</b> )
<b>SWRC0000</b>	Read Committed / HeldCursors=No (default on <b>DB2 UDB</b> for Windows and UNIX, and <b>DB2 for VM/VSE</b> )
SWRR0000	Repeatable Read / HeldCursors=No
SWRU0000	Read Uncommitted / HeldCursors=No
SWTS0000	Serializable / HeldCursors=No
<b>SWRC1000</b>	Read Committed / HeldCursors=Yes (default on <b>DB2 for OS/390 v5.1</b> and later)

Package Name	IsolationLevel and HeldCursors Setting
SWRR1000	Repeatable Read / HeldCursors=Yes
SWRU1000	Read Uncommitted / HeldCursors=Yes
SWTS1000	Serializable / HeldCursors=Yes

Table 8 through Table 10 show the packages that can be bound when using the KeepDynamic option, or if the BindRules keyword is set to BIND. These tables are only applicable for hosts running DB2 for OS/390 v5.1 and later.

**Table 8. Dynamic Packages on DB2 for OS/390 with KeepDynamic=Yes**

Package Name	IsolationLevel and HeldCursors Setting
SWNC4000	No Commit/Held Cursors=No
SWRC4000	Read Committed / HeldCursors=No
SWRR4000	Repeatable Read / HeldCursors=No
SWRU4000	Read Uncommitted / HeldCursors=No
SWTS4000	Serializable / HeldCursors=No
<b>SWRC5000</b>	Read Committed / HeldCursors=Yes (default on <b>DB2 for OS/390 v5.1</b> and later)
SWRR5000	Repeatable Read / HeldCursors=Yes
SWRU5000	Read Uncommitted / HeldCursors=Yes
SWTS5000	Serializable / HeldCursors=Yes



**Table 9. Dynamic Packages on DB2 for OS/390 with BindRules=BIND and KeepDynamic=No**

<b>Package Name</b>	<b>IsolationLevel and HeldCursors Setting</b>
SWRC2000	Read Committed / HeldCursors=No
SWRR2000	Repeatable Read / HeldCursors=No
SWRU2000	Read Uncommitted / HeldCursors=No
SWTS2000	Serializable / HeldCursors=No
SWRC3000	Read Committed / HeldCursors=Yes
SWRR3000	Repeatable Read / HeldCursors=Yes
SWRU3000	Read Uncommitted / HeldCursors=Yes
SWTS3000	Serializable / HeldCursors=Yes

**Table 10. Dynamic Packages on DB2 for OS/390 with BindRules=BIND and KeepDynamic=Yes**

<b>Package Name</b>	<b>IsolationLevel and HeldCursors Setting</b>
SWRC6000	Read Committed / HeldCursors=No
SWRR6000	Repeatable Read / HeldCursors=No
SWRU6000	Read Uncommitted / HeldCursors=No
SWTS6000	Serializable / HeldCursors=No
SWRC7000	Read Committed / HeldCursors=Yes

Package Name	IsolationLevel and HeldCursors Setting
SWRR7000	Repeatable Read / HeldCursors=Yes
SWRU7000	Read Uncommitted / HeldCursors=Yes
SWTS7000	Serializable / HeldCursors=Yes

## Static SQL Packages

If StarSQL users run applications that use static SQL, there must be a static SQL package for the SQL statements in the application on the host. If you do not know whether your applications use static SQL, consult the application developer.

There are two ways to bind a static SQL package—by recording it, or by manually binding it using the StarScribe Package Editor, which is available in the Windows-based version of StarSQL.

You can bind a static SQL package by enabling Static SQL Recording and Matching, and then connecting to the host and running the application. To configure Static SQL Recording and Matching from a UNIX computer, use the **restart** command. With Static SQL Recording and matching enabled, StarSQL will automatically bind the package and create a local static package file on the computer.

## Permissions for Packages

After the packages have been bound, StarSQL users must be granted permission to execute them. On DB2 for i, users must be granted \*USE permission on the packages, which can usually be done by the package owner. On other hosts, the DBA must grant StarSQL users EXECUTE or RUN permissions on the packages.

### Static and Catalog Packages

The StarSQL user executes applications that use the catalog package and any static SQL packages under the permissions of the package owner. No additional permissions are required to use these packages.

### Dynamic SQL Packages

On all hosts, except for DB2 for OS/390 v5.1 and later, for applications that use dynamic SQL, the StarSQL user needs explicit permissions to read (SELECT) and write (UPDATE/INSERT/DELETE) the columns and tables accessed by the application. The DBA needs to grant these permissions to “public” or to the various groups.

On DB2 for OS/390 v5.1 and later, the required permissions depend on the setting of the BindRules keyword. If the BindRules keyword is set to RUN, which is the default, the StarSQL user needs explicit permissions to read and write the columns and tables accessed by the application. If the BindRules option is set to BIND, the user has the permissions of the package owner, except that the following SQL statements cannot be executed regardless of the permissions of the package owner:

- SET CURRENT SQLID
- GRANT
- REVOKE
- ALTER
- CREATE
- DROP
- Any SQL statement that cannot be prepared as a dynamic SQL statement.

## Granting Use Permissions

To grant EXECUTE authority on the SQL packages, you can use the StarAdmin application if you have a Windows-based StarSQL computer, or you can execute SQL statements similar to those shown in the following sections.

### For DB2 for OS/390

Using SPUFI on the host or the SQL pass through mode of an ODBC-enabled application, execute the following SQL statements:

```
GRANT EXECUTE ON PACKAGE
STARSQL.SYSIBM, STARSQL.SWRC5000
TO PUBLIC
```

### For DB2 UDB (Windows and UNIX)

Using the DB2 Command Line Processor or the SQL pass through mode of an ODBC-enabled application, execute the following SQL statements:

```
GRANT EXECUTE ON PACKAGE
STARSQL.SYSCAT, STARSQL.SWRC0000
TO PUBLIC
```

## For DB2 for i

Using STRSQL (which is the interactive SQL interpreter, available in the Query Manager and SQL Development Kit for AS/400, Licensed Program 5769-ST1 or 5722-ST1), or using the SQL pass through mode of an ODBC-enabled application, execute the following SQL statements:

```
GRANT EXECUTE ON PACKAGE
  STARSQL.QSYS2, STARSQL.SWNC0000
  TO PUBLIC
```

In addition, you can use AS/400 authority commands to grant \*USE authority for all users (\*PUBLIC) to the library packages:

1. From a 5250 session, enter **WRKLIB** *<package library name>*.
2. Select Option 12 (work with objects).
3. Select Option 2 (edit authority on each object one at a time).
4. Change **PUBLIC \*EXCLUDE** to **PUBLIC \*USE**.

Another alternative is to use the GRTOBJAUT command from a 5250 session to grant \*USE authority for the library and EXECUTE authority for the packages to all users. The following example assumes that the package library is named "STARSQL."

```
GRTOBJAUT OBJ (STARSQL) OBJTYPE (*LIB) USER (*PUBLIC)
AUT (*USE)
GRTOBJAUT OBJ (STARSQL/*ALL) OBJTYPE (*ALL)
USER (*PUBLIC) AUT (*USE)
```

## For VM and VSE

For SQL/DS, use SQLDBA as the Package Collection if you want other users to have GRANT authority. Using any value other than SQLDBA for the Package Collection will restrict the ability to GRANT EXECUTE to other AUTHIDSs.

## AutoBind Option

The AutoBind configuration keyword forces SQL packages to be bound at connect time (SQLConnect or SQLDriverConnect). You can set AutoBind in the DSN definition (see [page 100](#)) or pass it in the connect string to *SQLDriverConnect()*.



---

### Caution

The use of AutoBind always causes binding to occur and adversely affects connect time performance.

---

The AutoBind option settings are described in [Table 11](#).

**Table 11. AutoBind Keyword Settings**

AutoBind Value	Description
0 (Default)	No auto-bind at connect time. The driver still binds packages on-the-fly if it does not find them on the server when it needs them.
1	Binds up to three packages at connect time, but uses NO REPLACE option on bind. If the packages already exist they will not be replaced. In the current version of StarSQL, AutoBind=1 ignores the NO REPLACE option and functions in the same manner as AutoBind=2.
2	Binds up to three packages at connect time and always replaces current packages, if there are any.
3	Binds the Package File (for static SQL) only at connect time, always replacing the current package if there is any.
Y	Same as 3.
N	Same as 0.

Regardless of the AutoBind setting, the StarSQL driver binds packages dynamically if it does not find them on the host at the time they are needed.

The three packages that may be bound by AutoBind=1 and AutoBind=2 are:

- the dynamic package for the current transaction isolation level
- the catalog package
- the static package file (a static SQL package file is bound only if static SQL matching is enabled for the data source)



# Manually Installing 32-bit StarSQL for UNIX

---

The procedures for manually installing StarSQL for UNIX vary depending on your operating system and whether you use an X Windows interface or a character-oriented terminal. The procedures in the following sections describe the typical methods for each supported variant of UNIX so you can choose which to use. If you need to remove the StarSQL for UNIX software, refer to ["Removing the StarSQL 32-bit Driver" on page 26](#).

It is possible to use the Linux version of StarSQL in a FreeBSD environment if your ODBC-enabled application is a Linux binary that runs in Linux binary compatibility mode. However, to run both the FreeBSD and Linux version of StarSQL software on the same computer, make sure that the `LD_LIBRARY_PATH` environment variable references only one version of StarSQL, and that the StarSQL library appears before the local library in the `LD_LIBRARY_PATH`.

## Installing StarSQL on AIX

You can use either the System Maintenance Interface Tool (SMIT) or the `installp` command to install the StarSQL software.

### Installation Using SMIT

Follow the instructions in this section if you want to use the SMIT to install the StarSQL software.

1. Logon to UNIX as root user.
2. Select the following options from the SMIT menus:
  - "Software Installation and Maintenance"
  - "Install and Update Software"
  - "Install/Update Software by Package Name"

The “Install New Software Products by Package Name” dialog box appears.

3. In the “\*INPUT device/directory for software” field, enter the directory name of the installable StarSQL package.
4. Click on OK.

A dialog box with a set of choices appears.

5. Enter **starsql** in the “SOFTWARE to install” field and click OK.

The “Install New Software Products by Package Name” dialog appears.

6. Make sure the “PREVIEW only?” field is set to “no” and click OK. Click OK again to verify your choice.

While SMIT installs the StarSQL software you will see an icon of a person running. When the installation is complete, the icon changes to a person holding up their arms in victory and OK appears.

7. Click the Done button to dismiss the installation dialog.
8. Edit the \$STARDIR/etc/unixodbc.ini file and replace %TARGET with the pathname to the installed StarSQL software (such as /usr/lpp/starsql).
9. Edit the \$STARDIR/etc/post\_install script and replace %TARGET with the pathname to the installed StarSQL software.
10. Enter the following command to run the “post\_install” script, which sets any environment variables that need to be specified and registers the StarSQL driver with the ODBC Driver Manager.

```
$STARDIR/etc/post_install
```

## Installation Using a Character-Oriented Terminal

An alternative to using SMIT is to install StarSQL from a character-oriented terminal using the **installp** command.

1. Logon to UNIX as root user.
2. Change to the directory containing the StarSQL installer. Typically the image is downloaded to /tmp/starsql.
3. Enter the following command:

```
# installp -a -d ./starsql starsql.base
```

4. Edit the \$STARDIR/etc/unixodbc.ini file and replace %TARGET with the pathname to the installed StarSQL software (such as /usr/lpp/starsql).



5. Edit the `$$STARDIR/etc/post_install` script and replace `%TARGET` with the pathname to the installed StarSQL software.
6. Enter the following command to run the “post\_install” script, which sets any environment variables that need to be specified and registers the StarSQL driver with the ODBC Driver Manager.

```
$$STARDIR/etc/post_install
```

## Installing StarSQL on FreeBSD

StarSQL for UNIX is distributed for installation on FreeBSD as a tar file of the `$$STARDIR` directory.

1. Logon to FreeBSD as `root` user.
2. Create an installation directory and change to that directory, such as:

```
# mkdir /usr/local/share/starsql
# cd /usr/local/share/starsql
```
3. Extract the contents of `starsql.tar`.

```
# tar xf /tmp/starsql.tar
```
4. Edit the `$$STARDIR/etc/unixodbc.ini` file and replace `%TARGET` with the pathname to the installed StarSQL software (such as `/usr/local/share/starsql`).
5. Edit the `$$STARDIR/etc/post_install` script and replace `%TARGET` with the pathname to the installed StarSQL software.
6. Enter the following command to run the “post\_install” script:

```
$$STARDIR/etc/post_install
```

## Installing StarSQL on HP-UX

To install the StarSQL software on an HP-UX computer you use the **swinstall** utility, which can be run in either interactive mode (using the X Windows or curses application interface) or non-interactive mode. The procedures for both modes are provided in the subsections below. Use the mode you prefer, and refer to the HP-UX Installation manual if you need detailed instructions about using the **swinstall** utility.

## Using swinstall in Interactive Mode

You can run swinstall in interactive mode to select which software bundle you want to install on the HP-UX computer. If the DISPLAY environment variable is defined, swinstall will run as an X Windows application. If DISPLAY is not set, swinstall runs as a curses application. You also can invoke swinstall in the interactive mode by selecting Software Management from the Systems Administration Manager (sam).

1. Logon to HP-UX as root user.
2. Copy the StarSQL installer to a local directory of the computer.
3. Enter **swinstall** at the command line.
4. During the selection phase you are prompted to specify the source of the software you want to install, such as:  

```
Install Software to Local Host
Source Depot Type: Local Directory
Source Depot Path: /opt/starsql/ (or the directory that contains
the extracted installer)
```
5. If you want to install StarSQL to a location other than the default /opt directory, select the StarSQL product and choose Change Target from the Actions menu.
6. Select the StarSQL product and choose Install from the Actions menu to add the StarSQL package.
7. After the StarSQL software is installed, edit the \$STARDIR/etc/unixodbc.ini file and replace %TARGET with the pathname to the installed StarSQL software (such as /opt/starsql).
8. Edit the \$STARDIR/etc/post\_install script and replace %TARGET with the pathname to the installed StarSQL software.
9. Enter the following command to run the “post\_install” script, which sets any environment variables that need to be specified and registers the StarSQL driver with the ODBC Driver Manager.

```
$STARDIR/etc/post_install
```

## Using swinstall in Non-Interactive Mode

To run swinstall in non-interactive mode you specify the -s parameter to provide the source depot name. The source depot name must be an absolute pathname of the tar file “starsql”.

1. Logon to HP-UX as `root` user.
2. Copy the StarSQL installer to a local directory of the computer.
3. Change to the directory where you extracted the StarSQL .tar file.
4. Enter the `swinstall` command, specifying the absolute pathname of the tar file “starsql” for the value of the `-s` parameter, such as shown in the following example:

```
# swinstall -s /opt/starsql/hp-ux/starsql
starsql.base
```

To install StarSQL to a location other than the default `/opt` directory, specify the different directory when you enter the `swinstall` command, such as:

```
# swinstall -s /opt/starsql/hp-ux/starsql
starsql.base @ /different_base_directory
```

5. After the StarSQL software is installed, edit the `$STARDIR/etc/unixodbc.ini` file and replace `%TARGET` with the pathname to the installed StarSQL software (such as `/opt/starsql`).
6. Edit the `$STARDIR/etc/post_install` script and replace `%TARGET` with the pathname to the installed StarSQL software.
7. Enter the following command to run the “`post_install`” script, which sets any environment variables that need to be specified and registers the StarSQL driver with the ODBC Driver Manager.

```
$STARDIR/etc/post_install
```

## Installing StarSQL on Linux

StarSQL for Linux is distributed for installation in RPM format and as a tar file. Follow the procedures in one of the following two subsections, depending on which installation method you prefer to use.

### RPM Installation

Some distributions of Linux include the RPM (RPM Package Manager) for installing software packages. The following procedures describe how to install the RPM format of the StarSQL distribution package.

1. Logon to Linux as `root` user.
2. Change to the directory that contains the StarSQL installer, typically `/tmp/starsql`.

3. Enter the following command to start the package manager, replacing the *<RPM file>* variable with the actual name of the StarSQL package (such as `starsql-5.51-1.i386.rpm`).  

```
# rpm -i <RPM file>
```
4. After the StarSQL software has been installed, edit the `$(STARDIR)/etc/unixodbc.ini` file and replace `%TARGET` with the pathname to the installed StarSQL software (such as `usr/share/starsql`).
5. Edit the `$(STARDIR)/etc/post_install` script and replace `%TARGET` with the pathname to the installed StarSQL software.
6. Enter the following command to run the “post\_install” script, which sets any environment variables that need to be specified and registers the StarSQL driver with the ODBC Driver Manager.  

```
$(STARDIR)/etc/post_install
```

## Installation Using tar

The StarSQL for Linux distribution is also available as a tar-based installer that does not require the RPM Package Manager. If your version of Linux does not include the Package Manager, install the tar version of StarSQL.

1. Logon to Linux as `root` user.
2. Create an installation directory and change to that directory, such as:  

```
# mkdir /usr/share/starsql  
# cd /usr/share/starsql
```
3. Extract the contents of `starsql.tar`.  

```
# tar xf /tmp/starsql.tar
```
4. After the tar file is extracted, edit the `$(STARDIR)/etc/unixodbc.ini` file and replace `%TARGET` with the pathname to the installed StarSQL software (such as `usr/share/starsql`).
5. Edit the `$(STARDIR)/etc/post_install` script and replace `%TARGET` with the pathname to the installed StarSQL software.
6. Enter the following command to run the “post\_install” script, which sets any environment variables that need to be specified and registers the StarSQL driver with the ODBC Driver Manager.  

```
$(STARDIR)/etc/post_install
```

## Installing StarSQL on Solaris

To install StarSQL for UNIX on a Solaris computer you can use either the X Windows interface or you can run the `pkgadd` command from a character-oriented terminal. Follow the procedures in one of the following subsections, depending on which method you prefer to use.

### Installation Using X Windows

1. Logon to Solaris as `root` user.
2. Run **admintool**.
3. From the Browse menu, select Software.  
It may take a couple of minutes to display a list of installed software.
4. From the Edit menu, select Add.  
The Set Source Media dialog appears.
5. Select Hard Disk for the software location.
6. Click on the Directory field and enter the location of the StarSQL installation package
7. Click OK.  
The Add Software dialog box appears.
8. Select StarSQL and click Customize.  
The Customize Installation dialog box appears.
9. Highlight the Installation Directory box, enter the location where you want to install StarSQL, and click OK.
10. After the installation is complete, exit **admintool**.
11. Edit the `$(STARDIR)/etc/unixodbc.ini` file and replace `%TARGET` with the pathname to the installed StarSQL software (such as `opt/starsql`).
12. Edit the `$(STARDIR)/etc/post_install` script and replace `%TARGET` with the pathname to the installed StarSQL software.
13. Enter the following command to run the “post\_install” script, which sets any environment variables that need to be specified and registers the StarSQL driver with the ODBC Driver Manager.

```
$(STARDIR)/etc/post_install
```

## Installation Using a Character-Oriented Terminal

1. Logon to Solaris as root user.
2. Run the pkgadd command.

```
pkgadd -d /tmp starsql
```

---

### Note

The above command examples install StarSQL to the default location, /opt/starsql. To specify an alternate location, use -a none on the pkgadd command line and you will be prompted for the location to install the product.

---

3. After StarSQL is installed, edit the \$STARDIR/etc/unixodbc.ini file and replace %TARGET with the pathname to the installed StarSQL software (such as opt/starsql).
4. Edit the \$STARDIR/etc/post\_install script and replace %TARGET with the pathname to the installed StarSQL software.
5. Enter the following command to run the “post\_install” script, which sets any environment variables that need to be specified and registers the StarSQL driver with the ODBC Driver Manager.

```
$STARDIR/etc/post_install
```

# Manually Installing 64-bit StarSQL for Linux

---

The procedures in this appendix describe how to manually install the 64-bit version of StarSQL for UNIX on a Linux-based computer. StarSQL for Linux is distributed in .rpm format for installation by the RPM Package Manager and as a tar file for installation using the tar command. Procedures are provided in this appendix for using either installation method. If you need to remove the StarSQL for UNIX software, refer to "[Removing StarSQL from a 64-bit Computer](#)" on page 29.

## Installing StarSQL on a Linux Computer

The steps in the sections below describe how to install the 64-bit version of StarSQL using either the Package Manager (RPM) or the tar command.

### Installation of StarSQL Using RPM

If you are running Red Hat Linux or another distribution that includes the Package Manager, you can use the RPM to install StarSQL.

1. Logon to Linux as `root` user.
2. Change to the directory that contains the StarSQL installer, typically `/tmp/starsql64`.
3. Enter the following command to start the package manager, replacing the `<RPM file>` variable with the actual name of the StarSQL package (such as `starsql64-5.51-1.x86_64.rpm`).

```
# rpm -i <RPM file>
```
4. After the StarSQL software has been installed, edit the `$/STARDIR/etc/unixodbc.ini` file and replace `%TARGET` with the pathname to the installed StarSQL software (such as `usr/share/starsql64`).

5. Edit the `$STARDIR/etc/post_install` script and replace `%TARGET` with the pathname to the installed StarSQL software.
6. Enter the following command to run the “post\_install” script, which sets any environment variables that need to be specified and registers the StarSQL driver with the ODBC Driver Manager.

```
$STARDIR/etc/post_install
```

## **Installation of StarSQL Using tar**

The StarSQL for Linux software also is available as a tar file that can be installed from a command line. If your Linux distribution does not include RPM, install the tar version of StarSQL.

1. Logon to Linux as `root` user.
2. Create an installation directory and change to that directory, such as:

```
# mkdir $STARDIR
# cd $STARDIR
```
3. Extract the contents of `starsql.tar`.

```
# tar xf /tmp/starsql64.tar
```
4. After the tar file is extracted, edit the `$STARDIR/etc/unixodbc.ini` file and replace `%TARGET` with the pathname to the installed StarSQL software (such as `usr/share/starsql64`).
5. Edit the `$STARDIR/etc/post_install` script and replace `%TARGET` with the pathname to the installed StarSQL software.
6. Enter the following command to run the “post\_install” script, which sets any environment variables that need to be specified and registers the StarSQL driver with the ODBC Driver Manager.

```
$STARDIR/etc/post_install
```



# StarSQL for UNIX Command Reference

---

This chapter provides a reference for using the following StarSQL for UNIX commands:

- [pkgviewer](#), page 82
- [restart](#), page 84
- [simpleconn](#), page 86
- [starlic-clientcfg](#), page 88
- [starping](#), page 90
- [trcstart](#), page 91
- [trcviewer](#), page 92

## pkgviewer

### Purpose

View the contents of a static SQL package file.

### Synopsis

```
pkgviewer [-s] [-f LogFile] PkgFile
```

### Options

<b>-s</b>	View only package information and a statement count.
<b>-f <i>LogFile</i></b>	Name of the output file. If not specified, the output is sent to stdout.
<b><i>PkgFile</i></b>	Name of the static SQL package file to be viewed.

### Description

View the contents of a static SQL file created with the StarSQL driver. The package file may include package information, static SQL statements, parameters, and result columns.

To edit the package file using the Windows-based StarScribe Package Editor, copy it to a Windows machine in binary mode. The file name should have an extension of **.pkg**.

### Example

```
$ pkgviewer -s demo.pkg  
Package Name: SWX79PKN (demo.pkg)  
Collection: ODBC3  
DSN:  
Server: DB2V8C  
Number Of Statements: 12  
MaxNumSections: 1000  
DefaultCCSID: 37  
IsolationLevel: Read Committed  
Explain: No  
Token: 1000216  
Clean: No  
NextStatementNumber: 13  
PackageOwnerID: STARGAZER
```

```
ProcedureOwnerID: STARPROCS  
DefaultQualifier: ???  
Version: ???  
  
Number of Statements: 12
```

## **See Also**

[restart](#), page 84

## recstart

### Purpose

Configures the StarSQL driver for static SQL recording and matching.

### Synopsis

```
recstart DSN [-disable] | [-match PkgFile] |  
[-capture [-pqd] PkgFile]
```

### Options

<i>DSN</i>	The data source for which recording or matching is being enabled or disabled.
<b>-disable</b>	Disables static SQL recording or matching.
<b>-match</b> <i>PkgFile</i>	Enables StarSQL to match static SQL statements in the application with static SQL statements in the specified client package file, and to use the static SQL on the host database whenever it encounters a match.
<b>-capture</b> [-pqd] <i>PkgFile</i>	Enables StarSQL to record static SQL statements in the application into the specified client package file. You can specify any combination of the following switches prior to the <i>PkgFile</i> name to control which statements are recorded: <ul style="list-style-type: none"><li><b>-p</b> Specifies that only static SQL statements with parameter markers be recorded.</li><li><b>-q</b> Specifies that only static SQL SELECT statements be recorded.</li><li><b>-d</b> Specifies that DDL statements be recorded.</li></ul>

### Description

Configures the StarSQL driver to enable and disable static SQL recording and matching for the specified data source. Use the **pkgviewer** command to view the contents of the package file.

## Example

The following example turns on recording for a data source named DSN1 into a package file named demo.pkg.

```
$ recstart DSN1 -capture demo.pkg
```

Static SQL recording has been enabled:

```
Data Source: DSN1
Package File: demo.pkg
Statements with parameter markers only: No
Include DDL statements: No
SELECT statements only: No
```

The following example turns off recording.

```
$ recstart DSN1 -disable
```

Static SQL matching and recording have been disabled:

```
Data Source: DSN1
```

## See Also

[pkgviewer](#), page 82

# simpleconn

## Purpose

Tests that StarSQL can use the unixODBC driver manager to connect to a specific data source on the host.

## Synopsis

```
simpleconn DSN UserId Password [#Connections]
```

## Options

<i>DSN</i>	The data source name to which you want to connect.
<i>UserId</i>	A valid user ID for a user account on the host.
<i>Password</i>	The valid password for the userid.
<i>#Connections</i>	The number of times to connect to the data source.

## Description

You can use the simpleconn command to test that StarSQL can use the unixODBC driver manager to successfully connect to a specified data source on the host. Upon successful connection, the simpleconn command returns the database name, platform, and version, and the driver name and version.

You must specify a username and password of an account that is valid on the host and has permission to access the database. (See ["Preparation Required for All Hosts" on page 45](#) for more information about user accounts and permissions.)

The #Connections option lets you specify how many connections you want to establish. The default is 1. If you specify more than one connection, the simpleconn program establishes that number of connections, and then releases them before the command returns.

This command invokes the simpconn example program included with StarSQL in the \$STARDIR/samples directory. The simpleconn program is compiled with unixODBC include files and libraries.

## Example

The following command example tests that StarSQL can establish two connections to a data source named DSN1 using the unixODBC driver manager.

```
$ simpleconn DSN1 staruser starpass 2
```

```
Connection #1
```

```
Driver Name: libswodbc.a
```

```
Driver Version: 5.50.nnnn
```

```
Database Name: DB2 Universal DataBase
```

```
Database Version: 08.021.0007
```

```
Connection #2
```

```
Driver Name: libswodbc.a
```

```
Driver Version: 5.50.nnnn
```

```
Database Name: DB2 Universal DataBase
```

```
Database Version: 08.021.0007
```

## See Also

[starping](#), page 90

## starlic-clientcfg

### Purpose

Configure and manage StarLicense client computer.

### Synopsis

```
starlic-clientcfg [options]
```

### Options

Some of the more commonly-used options for configuring a client computer are shown below. Note that the StarLicense Client Configuration Menu provides an easier method for managing StarLicense client-server connections. Refer to [page 117](#) for information about using the StarLicense Client Configuration Menu.

**client-loglevel-set** [0 | FFFFFFFF]

Setting the log level to 0 (zero) disables logging. Set the log level to FFFFFFFF (eight F's) to enable logging of activity between the client and license service. The log file is created in the /tmp directory with the filename format **starliccli.<processID>.log**.

**client-loglevel-show**

Displays the log level value, which indicates whether logging is enabled (FFFFFFF) or disabled (0).

**client-server-show**

Shows information about the license server(s) the client is configured to use.

**client-server-add** <product\_ID> <hostname> <port> [PRI|SEC]

Configures the client to check out a license for the specified StarQuest product on the specified host. The product ID must match the product ID specified on the StarLicense server. The *hostname* can be the DNS name or TCP/IP address of the StarLicense server and the *port* variable is the number of the port on which the server is listening for license requests. Specify the option PRI to add the server as a Primary server, or SEC to add it as a Secondary server.

**client-server-remove** <product\_ID> <hostname> <port> [PRI | SEC]

Removes the specified StarLicense server from the client license configuration.



## Description

Use the **client-server** options to configure the client computer to use one or more StarLicense servers. The StarLicense server software can be co-located on the same computer as the StarSQL driver, or it can be installed on a network server.

## Example

The following example configures the computer to checkout StarSQL licenses from a StarLicense server named **starlic** that is listening for license requests on port 4999.

```
# ./starlic-clientcfg client-server-add SQ starlic 4999 PRI
```

The following example configures the client computer to use a StarLicense server named **starlic2** as a Secondary server if the Primary license server cannot satisfy the license request. The starlic2 server listens for license requests on port 5001.

```
# ./starlic-clientcfg client-server-add SQ starlic2 5001  
SEC
```

## See Also

["Licensing StarQuest Products" on page 31](#)

["StarLicense Client Configuration Utility" on page 117](#)

## starping

### Purpose

Tests the StarSQL connection to a specific data source.

### Synopsis

```
starping [DSN UserID Password]
```

### Options

<i>DSN</i>	The data source to which you want to connect.
<i>UserID</i>	A valid user ID for an account on the host.
<i>Password</i>	The valid password for the user ID.

### Description

The **starping** command connects to the specified DSN using a user ID and password. If you enter the command without the required parameters, the command prompts for the data source name, user ID, and password before it attempts the connection.

After a successful connection, the command returns “Connection Succeeded!” and the data source name (DSN) and database platform version number.

### Example

```
starping DSN1 staruser starpass
```

### See Also

[simpleconn](#), page 86

# trcstart

## Purpose

Starts and ends DRDA tracing.

## Synopsis

```
trcstart {-off|-on TraceFile}
```

## Options

**-off** Turns off the DRDA trace recording currently in progress.  
**-on TraceFile** Turns on DRDA trace recording to the specified trace file.

## Description

The **trcstart** command allows a user to capture the DRDA data streams that the StarSQL driver is sending and receiving. Use the **trviewer** command to view the contents of the trace file.

To enable DRDA tracing the `.swodbc.ini` file must be in the user's home directory and the user must have permission to write to the file.

## Examples

To copy `swodbc.ini` to the user home directory and make it writable:

```
$ cp $STARSQL/etc/swodbc.ini .swodbc.ini
$ ls -l .swodbc.ini
$ chmod 644 .swodbc.ini
```

To enable DRDA tracing:

```
$ trcstart -on /tmp/mytrace.sq
```

To turn off tracing:

```
$ trcstart -off
```

## See Also

[trviewer](#), page 92

## trcviewer

### Purpose

View the contents of a DRDA trace file.

### Synopsis

```
trcviewer [-a] [-e] [-h] | [-f LogFile] TraceFile
```

### Options

<b>-a</b>	Displays the trace information in ASCII format.
<b>-e</b>	Displays the trace information in EBCDIC format.
<b>-h</b>	Displays the trace information in hexadecimal format.
<b>-f</b> <i>LogFile</i>	The output filename (sent to stdout if no filename is specified).
<i>TraceFile</i>	The DRDA trace file to be viewed

### Description

The **trcviewer** command displays the contents of a DRDA trace file that is created by enabling DRDA tracing using the **trcstart** command. The trace file includes an analysis of command chains, code points, parameters, and SQL data rows. If you specify any of the format options (-a, -e, or -h), the data block also is displayed in ASCII, EBCDIC, or hexadecimal format.

Note that the StarSQL Trace Viewer utility for Windows provides more display options than the **trcviewer** UNIX command equivalent provides. To view the trace file with the Windows-based StarSQL Trace Viewer application, copy it to a Windows computer in binary mode. The Windows-based Trace Viewer application by default looks for trace files with an extension of **.sqd**, although you have the option of opening a trace file with any name.

### Example

```
trcviewer -f output.txt /tmp/mytrace.sqd
```

### See Also

[trcstart](#), page 91

# Customizing the StarSQL for UNIX Configuration

---

You can control how the StarSQL driver interacts with a data source by modifying the data source configuration settings. There are three primary methods for defining an ODBC data source:

- use the unixODBC ODBC Data Source Administrator, as described in ["Customizing the unixODBC Driver Manager Configuration" on page 34](#)
- edit the System DSN file (odbc.ini) or the User DSN file (.odbc.ini)
- create a File DSN

For a StarSQL data source you can specify Global Settings, which determine how the StarSQL driver behaves with all data sources, and you can define specific data source settings, which control how StarSQL interacts with a particular Data Source Name (DSN).

## Setting Global DSN Parameters

The Global Settings are defined in a file named `swodbc.ini` (or `.swodbc64.ini` on a 64-bit computer). The `swodbc.ini` file can be located in a shared network location with a symbolic link from the `$HOME` directory for client computers to find it, or a personal copy can be maintained in the `$HOME` directory of a 32-bit client computer (`.swodbc.ini`) or a 64-bit client computer (`.swodbc64.ini`).

The global DSN parameters are described within a stanza of the `swodbc.ini` configuration file and affect all DSNs. Within each stanza you specify keywords and values in the format shown below:

```
Keyword=value
```

The next section provides details about each configuration keyword you can specify in the `.swodbc.ini` global settings file.

## Global DSN Keywords

For most users, the default values for the global data source settings, which are stored in the `.swodbc.ini` file, are adequate. However, you can implement special behavior by modifying the global settings.

### AlwUpd

All database transactions are read/write by default (`AlwUpd=Yes`). Setting `AlwUpd=No` restricts the driver to read-only transactions.

You also can control whether transactions can be updated with the specific DSN configuration `ReadOnly` keyword (see "[ReadOnly](#)" on page 112). If both the `AlwUpd` global setting and the `ReadOnly` DSN setting are specified, the StarSQL driver uses the most restrictive setting. A global setting that allows read/write transactions (`AlwUpd=Yes`) can be overridden by a specific DSN setting that allows just read-only transactions (`ReadOnly=Yes`). Likewise, a transaction to a data source is restricted to read-only if the global setting restricts the driver to read-only (`AlwUpd=No`), even if the specific DSN configuration allows read/write (`ReadOnly=No`).

### AutoTypDefOvr

The `AutoTypDefOvr` setting stores the Coded Character Set Identifiers (CCSIDs) that the DB2 host provides to StarSQL upon connection. After the connection with the host is established, StarSQL uses the CCSID values specified for `AutoTypDefOvr` to send SQL statement and parameter data to the host unless the `TypDefOvr` value specifies to use different CCSIDs. The `TypDefOvr` setting takes precedence over values specified in the `AutoTypDefOvr` setting. However, if the `AutoTypDefOvr` CCSID values are appropriate, `TypDefOvr` values do not need to be sent with each DRDA request.

The `AutoTypDefOvr` keyword value consists of three comma-separated CCSID values for the single-byte character set (SBCS), the double-byte character set (DBCS), and the mixed-byte character set (MBCS), respectively.

`<SBCS CCSID>,<DBCS CCSID>,<MBCS CCSID>`

If the host returns a different CCSID than you want to use for the connection, or you want to override a multi-byte client code page with a MBCS CCSID, set the `TypDefOvr` to send a particular CCSID value with each DRDA request. (See "[TypDefOvr](#)" on page 112 for more information about the `TypDefOvr` setting.)

## CacheUID

If you set CacheUID=Yes (the default), the username to connect using StarSQL is displayed in the connect dialog and the user only needs to enter their password. Set CacheUID=No to require users to enter both their username and their password to connect to the DB2 host using StarSQL.

## Explain

DB2 provides an explain facility that provides detailed information about the access plan and environment of static or dynamic SQL statements. The captured information can help administrators understand how individual SQL statements are executed so the statement and database configuration can be tuned for performance. You can use a command-line tool or DB2 Visual Explain to display explain information.

StarSQL v5.4 supports a new keyword in the `.swodbc.ini` initialization file that binds all the StarSQL packages with the EXPLAIN bind option set to ALL. This includes dynamic SQL packages that are bound with StarAdmin, packages that are created and bound as needed by the StarSQL driver, and static SQL packages that are created with the StarScribe Package Editor.

To enable the EXPLAIN bind option for all packages that are bound by StarSQL, add the following line to the [Defaults] section of the `.swodbc.ini` file:

```
[Defaults]
Explain=All
```

After you modify the initialization file to include the Explain=All statement, the explain information for packages that are subsequently bound by StarSQL can be displayed.

## Setting Up System and User DSN Files

System DSN settings are defined in `odbc.ini`, and user DSN settings are defined in `$HOME/.odbc.ini`. The system DSN file uses the same format and keywords as for a user DSN file. The following procedures describe how to set up and edit the sample `.odbc.ini` file included in the StarSQL distribution to create a system or user DSN file with the required settings.

1. To create a system DSN, edit the `odbc.ini` file. The location of the file varies depending on which version of UNIX the computer is running, but is typically located in `/etc/odbc` or `/usr/local/etc`.

To create a user DSN, copy the `odbc.ini` file from `$STARDIR/etc` to your home directory and rename it `.odbc.ini` (preceded with a period):

```
$ cp $STARDIR/etc/odbc.ini $HOME/.odbc.ini
```

2. Make sure the system or user initialization file can be written to.  
To change the permissions for the system DSN file:
 

```
$ chmod 644 $STARDIR/etc/odbc.ini
```

 To change permissions for the user DSN file:
 

```
$ chmod 644 $HOME/.odbc.ini
```
3. Using a text editor such as vi or emacs, edit the required keywords, described in [Table 12](#), with values appropriate to your environment. You can further customize the DSN configuration with any of the keywords and values described in "[Sample DSN File](#)" on [page 97](#).

**Table 12. Required Data Source Configuration Parameters**

Keyword	Description
Driver=	If you install StarSQL to a location other than the default directory, specify the fully qualified path to the driver. You also can specify the name of the driver as it appears in the section head of the <code>odbcinst.ini</code> file, such as <code>Driver=StarSQL</code> or <code>Driver=StarSQL64</code> , to automatically use the location specified in the driver manager repository ( <a href="#">Figure 3 on page 35</a> shows an example <code>odbcinst.ini</code> file).
HostName=	The host name refers to the name of the DB2 host system. Set the <code>HostName</code> field to either a TCP/IP host name (such as <code>host5.mydomain.com</code> ) or a static IP address in dotted decimal notation (such as <code>198.147.235.1</code> ).
PkgColID=	The SQL Package Collection ID indicates the location on the DB2 host of the packages required by StarSQL to execute Dynamic SQL.  On DB2 for i, the SQL Package Collection ID is the name of the collection or library that contains the packages. On all other platforms, the Package Collection ID is the name of the virtual collection associated with these packages.  If necessary, obtain the package collection ID from your Database Administrator.



Keyword	Description
Port=	The Port designates the IP port on which the DB2 host is listening for incoming TCP/IP connection requests. The default port for DRDA communications is 446.
Server=	Enter the name of the database server that will be accessed through this data source. You may need to obtain the database server name from the Database Administrator.  The database server name is known by different names depending on the DB2 host implementation. On DB2 for OS/390 it is called the location name, on DB2 for i it is called the relational database name (RDB), and on DB2 UDB for Windows and UNIX, it is the name of the database.

The next section provides details about the configuration keywords you can specify in the `odbc.ini` system DSN file or the `.odbc.ini` user DSN file to customize the behavior of the StarSQL driver. The information for a particular DSN is described within a stanza of the configuration file. Within each stanza you specify keywords and values for the DSN in the format shown below:

```
Keyword=value
```

## Sample DSN File

Specific data source settings, which are stored in `$HOME/.odbc.ini` for user DSNs and in `$STARDIR/etc/odbc.ini` for system DSNs, control how StarSQL interacts with a particular data source. Below is a sample `.odbc.ini` file that provides a DSN definition for a data source named `SAMPLEDSN` that is accessed from an AIX computer using StarSQL for UNIX. The parameters listed at the beginning of the `[SAMPLEDSN]` section in **bold** are the minimum required for connectivity and are described in the table "[Required Data Source Configuration Parameters](#)" on page 96.

```
[ODBC Data Sources]
SAMPLEDSN=StarSQL

[SAMPLEDSN]
Driver=/usr/lpp/starsql/lib/libSWODBC.a
HostName=host.domain.com
PkgColId=STARSQL
Port=446
Server=RDBNAME
Accounting= " "
```

## Customizing the StarSQL for UNIX Configuration

```
AllowSynonyms=Yes
AutoBind=No
AutoSqlSet=D
AutoTypDefOvr=1252, 0, 0
BinaryCCSID=37
BindRules=Run
CacheUID=Yes
Capture=No
CaptureDDL=No
CaptureQryOnly=No
CatFilters='NAME1', 'NAME2'
CatQual=SYSIBM
CharacterSubstitution=Warning
CmdBufSiz=32767
CreateTable=IN DATABASE DSNDB04
CustomizePrdid=
DefaultQualifier=STARUSER
Descriptions=Sample DSN Configuration
ExpirationWarning=0
FetchAhead=Yes
FoldUIDPWD=No
GetInfoCatalogUsage=
HeldCursors=Default
IncludeSynonyms=Yes
IsolationLevel=<default>
KeepDynamic=Yes
LiteralConversion=Yes
LongStrParams=No
MaxRows=
OverrideCodeset=1252, S, Windows-1252, MYCODESETNAME
PackageFile=
ParameterMarkersOnly=No
PasswordProc=DRDA11DE
PkgOwnID=
QryBufSiz=32767
ReadOnly=No
SpecialColumns=Yes
StrictParsing=Yes
TypDefOvr=1252, 0, 0
UseDSCRDBTBL=Yes
UseDynamicCatalogSQL=Yes
UseEncryption=Any
UseJumboPackages=No
UseStaticMatch=No
UseSYSDUMMYAEU=Yes
```

The following sections describe the keywords you can specify in the DSN definition should you need to further customize how the StarSQL driver connects to a data source.

## System and User DSN Keywords

For most users, the default values that StarSQL uses for connecting to a data source are adequate. However, you can implement special behavior by adding or modifying the keywords described in this section.

### Accounting

OS/390 allows you to specify an accounting string for charging back the mainframe resources that particular users consume while connected to DB2 databases through StarSQL. The accounting string is passed in the PRDDTA parameter of the ACCRDB DRDA command. It is stored as an accounting record on the mainframe.

An accounting string contains a system-generated prefix and a user-generated suffix, which together provide the information needed to associate resource usage with a user's access for the purpose of charge-back accounting. The value you specify for the Accounting keyword provides the user-defined suffix of the full accounting string.

The accounting string prefix consists of 56 bytes generated by StarSQL. The fields are right-padded and are described in [Table 13](#).

**Table 13. Accounting String Prefix**

Field	Description
<i>acct_str_len</i>	1 byte, length of the accounting string minus 1 in hexadecimal format
<i>client_prdid</i>	8 bytes, product ID for the driver
<i>client_platform</i>	18 bytes, client operating system
<i>client_appl_name</i>	20 bytes, client application
<i>client_authid</i>	8 bytes, authorization id of the StarSQL user
<i>suffix_len</i>	1 byte, length of the accounting sting suffix in hexadecimal format

The user-defined suffix is used to further refine the accounting record. The user enters the suffix in the accounting string field in the data source setup. In the following example accounting string, the user-supplied suffix is SALES:

```
"x'3C'STARSQL2UNIX mypgm BARNES x'05'SALES"
```

If there is no value specified for the Accounting keyword, the accounting string will have a null suffix, as shown in the following example:

```
"x'3C'STARSQL2UNIX mypgm BARNES"
```

For information about interpreting the accounting record on DB2, see the IBM documentation for your version of DB2 for OS/390 (see "[Related IBM Reference Documentation](#)" on page 17 for publication numbers).

## AllowSynonyms

The default behavior of StarSQL is to retrieve index information for synonyms from DB2 for z/OS. To disable synonym support, enter AllowSynonyms=No in the data source configuration. If you set AllowSynonyms to No, StarSQL uses the default qualifier to qualify unqualified table references in catalog calls, such as with SQLStatistics().

## AutoBind

The AutoBind keyword forces SQL packages to be bound at connect time. Normally the use of AutoBind is not recommended, because it always causes binding to occur and thus adversely affects connect time performance.

The AutoBind option settings are described in [Table 14](#).

**Table 14. AutoBind Keyword Settings**

AutoBind Value	Description
0 (Default)	No auto-bind at connect time. The driver still binds packages on-the-fly if it does not find them on the server when it needs them.
1	Binds up to three packages at connect time, but uses NO REPLACE option on bind. If the packages already exist they will not be replaced. In the current version of StarSQL, AutoBind=1 ignores the NO REPLACE option and functions in the same manner as AutoBind=2.

<b>AutoBind Value</b>	<b>Description</b>
2	Binds up to three packages at connect time and always replaces current packages, if there are any.
3	Binds the Package File (for static SQL) only at connect time, always replacing the current package if there is any.
Y	Same as 3.
N	Same as 0.

Regardless of the AutoBind setting, the driver still binds packages dynamically if it does not find them on the host at the time they are needed. The three packages that may be bound by 1 and 2 are: the dynamic package for the current transaction isolation level, the catalog package, and the static package file.

## AutoSqlSet

Some DBMS systems accept information that identifies the client connection---user ID, computer name, and application name. Set AutoSqlSet to No if the host does not support this functionality or you do not want to send this client connection information to the DBMS.

If StarSQL detects that the host supports the client connection information, it automatically sets the AutoSqlSet setting to Yes. Set AutoSqlSet to D (for "disabled") if you do not want StarSQL to change the AutoSqlSet setting to Yes for hosts that accept client connection information.

## AutoTypDefOvr

You can set a value for AutoTypDefOvr in the global DSN `.swodbc.ini` file, for a system DSN in the `odbc.ini` file, or for a user DSN in the `.odbc.ini` file. The AutoTypDefOvr setting stores the Coded Character Set Identifiers (CCSIDs) that the DB2 host provides to StarSQL upon connection.

After the connection with the host is established, StarSQL uses the CCSID values specified for AutoTypDefOvr to send SQL statement and parameter data to the host unless the TypDefOvr value specifies to use different CCSIDs. The TypDefOvr setting takes precedence over values specified in the AutoTypDefOvr setting. However, if the AutoTypDefOvr CCSID values are appropriate, TypDefOvr values do not need to be sent with each DRDA request.

The AutoTypDefOvr keyword value consists of three comma-separated CCSID values for the single-byte character set (SBCS), the double-byte character set (DBCS), and the mixed-byte character set (MBCS), respectively.

`<SBCS CCSID>, <DBCS CCSID>, <MBCS CCSID>`

If the host returns a different CCSID than you want to use for the connection, or you want to override a multi-byte client code page with a MBCS CCSID, set the TypDefOvr to send a particular CCSID value with each DRDA request. (See "[TypDefOvr](#)" on page 112 for more information about the TypDefOvr setting.)

### BinaryCCSID

If you are experiencing problems with character data being returned as binary data from DB2 for i, specify the CCSID for the character set you want to use in the BinaryCCSID field. This instructs StarSQL to treat binary data as character data using the specified CCSID. For the complete list of supported CCSIDs, refer to the “StarSQL Character Conversion and National Language Support” document in the StarSQL technical documents of the StarQuest Web site.

### BindRules

The default value for the BindRules setting is RUN. This setting applies only to applications that use dynamic SQL on a data source on DB2 for z/OS. Set this option to RUN or BIND as desired.

On DB2 for z/OS, the permissions required to run applications that use dynamic SQL depend on the value of the BindRules option. If it is set to RUN, the StarSQL user needs to have explicit permissions granted by the administrator to read and write the columns and tables accessed by the application. If it is set to BIND, the StarSQL user executes with the permissions of the owner of the dynamic SQL package.

If the option is set to BIND, the following SQL statements cannot be executed, regardless of the actual permissions of the package owner:

- SET CURRENT SQLID
- GRANT
- REVOKE
- ALTER
- CREATE
- DROP
- Any SQL statement that cannot be prepared as dynamic SQL

## CacheUID

If you set CacheUID=Yes (the default), the username to connect using StarSQL is displayed in the connect dialog and the user only needs to enter their password. Set CacheUID=No to require users to enter both their username and their password to connect to the DB2 host using StarSQL.

## Capture

The default value for the Capture keyword is No, which prevents recording static SQL to a package file. Set Capture=Yes to enable static SQL recording to a package file. To limit recording to statements with parameter markers, also specify the ParameterMarkersOnly keyword (see [page 111](#)).

If you set Capture=Yes, you also must specify a value for the PackageFile keyword (see [page 110](#)).

Note that the Capture keyword does not affect recording of DDL statements. To record DDL statements, use CaptureDDL.

## CaptureDDL

Set CaptureDDL to Yes to enable recording of DDL statements to a package file. Set to No to disable recording of DDL statements. To limit recording to statements with parameter markers, also specify the ParameterMarkersOnly keyword (see [page 111](#)).

If you set CaptureDDL=Yes, you also must specify a value for the PackageFile keyword (see [page 110](#)).

## CaptureQryOnly

Set CaptureQryOnly to Yes to record only SQL SELECT statements to a package file. Set CaptureQryOnly=No if you do not want to restrict recording to SELECT statements. To record only SELECT statements that have parameter markers, also enable the ParameterMarkersOnly keyword (see [page 111](#)).

To enable recording of all static SQL in an application, except for DDL statements, use the Capture setting. To record DDL statements, use CaptureDDL.

If you set CaptureQryOnly=Yes, you also must specify a value for the PackageFile keyword (see [page 110](#)).

## CatFilters

The CatFilters setting restricts the amount of data retrieved by ODBC catalog functions when a NULL or '%' is passed as the schema name. Set CatFilters to the name of a particular library or collection to limit the amount of data retrieved from the specified

library or collection when a catalog function is executed. You can specify a maximum of 10 names for filtering the data. The names must be enclosed in single quotation marks and separated by a comma (such as 'name1', 'name2', 'name3', ...).

This feature is supported by the following DB2 hosts:

- DB2 for z/OS (DB2 for OS/390)
- DB2 for VM and VSE
- DB2 for i
- DB2 Universal Database (UDB)

### CatQual

Set the catalog qualifier to the location of the DB2 system tables that StarSQL uses when an application calls an ODBC catalog function. Set the catalog qualifier to SYSIBM for DB2 for z/OS, SYSCAT for DB2 UDB, or QSYS2 for DB2 for i (OS/400).

### CharacterSubstitution

If one or more characters cannot be converted between the host and client character sets, StarSQL substitutes a question mark, ?, in place of the character(s). The CharacterSubstitution parameter determines whether a warning is generated when characters are replaced (parameter value of Warning), or the characters are substituted without generating a warning (parameter value of Silent).

### CmdBufSiz

The size of the command buffer can be between 512 and 32767 bytes. The default setting is 32767 (32K). A large command buffer can improve performance for inserts and updates in which a single row contains more than 400 bytes of data.

### CreateTable

Set the CreateTable string to contain a directive that you want to append to SQL CREATE TABLE statements. The directive indicates where you want the table to be created.

For example, if **CreateTable=IN DATABASE DSNTTEST** and an application passes the following statement:

```
create table test (num integer)
```

the StarSQL driver passes the statement to DB2 as:

```
create table test (num integer) IN DATABASE DSNTTEST
```



Valid values for the CreateTable keyword depend on the host where the data source is located, as shown in [Table 15](#).

**Table 15. Supported CreateTable Directives**

DB2 Host	Valid Table Creation Directives	Comment
DB2 for z/OS	<b>IN DATABASE</b> <i>DATABASE_NAME</i> <b>IN</b> <i>DATABASE_NAME.TABLESPACE</i>	
DB2 for i	<b>IN NODE</b> <i>GROUP_NAME</i>	A node group name is a qualified or unqualified name that designates a nodegroup, which is a group of DB2 for i systems across which a table is distributed.
DB2 for VM and VSE	<b>IN</b> <i>DBSPACE</i>	
DB2 UDB	<b>IN</b> <i>TABLESPACE_NAME</i>	A tablespace name is a long qualifier that identifies a distinct table space that is described in the DB2 catalog.

## CustomizePrdid

The CustomizePrdid parameter customizes the behavior of a DB2 for z/OS host that is using a double-byte character set (DBCS). DB2 for z/OS does not support the ESCAPE clause in SQL syntax when the LIKE argument uses a mixed-byte character set.

When the CustomizePrdid parameter is set to M (for Mixed), StarSQL will strip the backslash escape character (\) from both the catalog SQL that is used to bind catalog packages and from dynamic catalog SQL calls. The CustomizePrdid parameter must be enabled prior to binding packages, and it must be enabled in the DSN that is used to connect to the host from the StarSQL client.

The escape character precedes a special character, such as the underscore (\_) character, to treat the character as a literal. To work with tables that include an underscore in the name, such as MY\_TABLE, create an ALIAS, SYNONYM, or VIEW for the table with a name that includes no special characters instead of working directly with the table.

## DefaultQualifier

Set DefaultQualifier to a value that you want to use to qualify all unqualified SQL statements. On DB2 for i, this qualifier refers to an AS/400 library. On other DB2 hosts it refers to an Owner or Authorization Identifier.

## Description

This is an optional keyword to provide a textual description of the data source.

## ExpirationWarning

The password ExpirationWarning keyword applies only to the Stored Procedures method for password management. If you are using this method, you can enter an integer greater than 0 (zero) to indicate the number of days prior to password expiration that the user will be warned. The default value, 0 (zero), means that the user will not be warned of password expiration.

## FetchAhead

FetchAhead is a means of retrieving data from the host before the application needs it. This data is fetched from the host, stored in the driver, and returned to the application on request. This improves performance by reducing delays in receiving data from the host. The default for FetchAhead is Yes.

## FoldUIDPWD

Set FoldUIDPWD to Yes if you want StarSQL to force the UserID and Password to uppercase letters if the user is connecting using Microsoft SNA. When FoldUIDPWD is set to No, user IDs and passwords are sent in the case they were entered.

## GetInfoCatalogUsage

Applications can call SQLGetInfo with the SQL\_CATALOG\_USAGE option to determine the classes of SQL statements in which catalog and schema names can be used to qualify objects of SQL statements. Setting the GetInfoCatalogUsage parameter of the StarSQL driver allows you to customize the bitmask that is returned by SQLGetInfo() for SQL\_CATALOG\_USAGE.

Set GetInfoCatalogUsage to 0 to force StarSQL to report that it does not support three-part naming (no Catalog). This may be necessary in some situations, such as when using StarSQL to perform queries to a SQL Server linked server. If GetInfoCatalogUsage contains an empty string or -1, the normal bitmask is returned.

## HeldCursors

You can set the HeldCursors keyword to No, Yes, or Default. Yes enables held cursors. No disables held cursors. The behavior of held cursors when the value is Default depends on the host platform, as shown in [Table 16](#). If the host does not support held cursors, the HeldCursors keyword is ignored.

**Table 16. Behavior of HeldCursors=Default by Host**

Host	Behavior if HeldCursors=Default
DB2 for z/OS and OS/390	Enables held cursors (equivalent of HeldCursors=Yes)
DB2 for i and DB2 UDB	Disables held cursors (equivalent of HeldCursors=No)

Normally, a cursor is closed when its transaction commits. If the HeldCursors keyword is enabled, the cursor remains open after the commit. This allows an application to fetch rows from a result set, commit the transaction, and then continue fetching additional rows on the same result set.

Using held cursors, an application can commit immediately after opening the result set to allow locks normally needed to maintain a prepared statement to be freed early. If the application is in auto-commit mode, StarSQL automatically issues a commit at the time the result set is opened and at the time the result set is closed.

An application that uses held cursors should turn off `SQL_AUTOCOMMIT_MODE` before executing a prepared statement multiple times. Otherwise, StarSQL prepares the SQL statement after each `SQLExecute`, resulting in slower performance.

Before enabling HeldCursors, manually delete any existing SQL package used for catalog information on DB2. This is the package with the same name as the catalog schema. A new SQL package will be created the next time you establish a connection using the updated StarSQL driver.

If you are upgrading from a previous version of StarSQL, you may need to delete and rebind the SQL catalog package if you change the HeldCursors setting.

## IncludeSynonyms

IncludeSynonyms support is enabled by default. Set IncludeSynonyms to No if you do not want to support SYNONYMS in catalog calls. When AllowSynonyms or IncludeSynonyms is disabled, synonyms are not returned in results and are not found when passed as input parameters. When AllowSynonyms or IncludeSynonyms is set to Yes, synonyms are supported, as both input and output, in catalog calls.

## IsolationLevel

IsolationLevel refers to the degree of concurrency allowed when multiple transactions try to access the same data. You can set the isolation levels for StarSQL using an integer that represents the level, as described in [Table 17](#). The StarSQL driver uses a default isolation level of 0 for the AS/400 host and a default of 2 for the other supported hosts.

**Table 17. IsolationLevel Values**

IsolationLevel Value	Description
<Default>	The default is dependent on the host platform. On DB2 for i, the default is None. For most host platforms, the default is Read Committed, which allows the StarSQL driver to access non-journalled files.
0	<b>None:</b> No isolation of concurrent application processes.
1	<b>Read_Uncommitted:</b> For a SELECT INTO, FETCH with a read-only cursor, subquery, or subselect used in an INSERT statement, allows any row read during the unit of work to be changed by other application processes and any row changed by another application process to be read even if the change has not been committed by that application process. For other operations, behaves like Read Committed.
2	<b>Read_Committed:</b> Similar to Repeatable Read in that a unit of work cannot be changed by another process until it has completed, and a row changed by another application process cannot be read until it has been committed. However, with Read Committed, an application process that executes the same query more than once may see newly committed rows inserted by another application process that were not present in the original read.

IsolationLevel Value	Description
4	<b>Repeatable_Read:</b> Ensures that any row read during a unit of work can not be changed by other application processes until the unit of work has completed. Also ensures that any row changed by another application process cannot be read until it has been committed. At this level, a process is completely isolated from the effects of concurrent application processes.
8	<b>Serializable:</b> Ensures that any row changed by another application process cannot be read until it is committed by that application process. Ensures only that the current row of every updateable cursor is not changed by other application processes. Rows read during a unit of work can be changed by other application processes.

## KeepDynamic

The KeepDynamic keyword enables or disables optimization for prepared statements by tuning the KeepDynamic bind option (DB2 for z/OS and DB2 UDB). The default for KeepDynamic is Yes.

## LiteralConversion

Set LiteralConversion to Yes to enable literal conversion, which causes StarSQL to substitute parameter markers for literals when doing static matching.

## LongStrParams

StarSQL v5 adds support for DB2 LOB (large object) data types. The LongStrParams setting provides backwards compatibility to use StarSQL v5 with hosts that do not support LOBs and applications that rely on the previous ODBC data types.

As shown in [Table 5 on page 22](#), StarSQL v5 changes the mapping for DB2 long strings—they are no longer differentiated from short strings:

The new data type mappings affect ODBC catalog query results. For example, SQLColumns for a DB2 LONG VARCHAR now returns ODBC type SQL\_VARCHAR instead of SQL\_LONGVARCHAR. The new mappings also affect parameters for SQL statements. If an application binds a parameter as SQL type SQL\_LONGVARCHAR, it is sent as a DB2 CLOB instead of as a DB2 VARCHAR string.

When LongStrParams is set to Yes, then SQL\_LONGVARCHAR is sent as VARCHAR instead of CLOB, and SQL\_LONGVARBINARY is sent as VARCHAR FOR BIT DATA instead of BLOB. This LongStrParams setting does not affect the types returned for result set columns, or for types returned by catalog queries.

When StarSQL v5 connects to an older version of DB2 that does not support LOB data types, the LongStrParams data source setting is forced to Yes and SQL\_LONGVARCHAR and SQL\_LONGVARBINARY parameters are sent as strings. Catalog queries, however, will not return SQL\_LONGVARCHAR or SQL\_LONGVARBINARY for long string types.

### MaxRows

Set the MaxRows keyword to the maximum number of rows that can be returned by a result set. Set this value to limit the number of rows returned in queries that do not call SQLSetStmtOption with the SQL\_MAX\_ROWS parameter.

If you do not want to set an upper limit, enter zero or leave the field blank.

### OverrideCodeset

The OverrideCodeset parameter allows you to override the name of the local codeset that StarSQL uses to convert outbound character data. The codeset names are defined in the library that contains the conversion routines.

StarSQL obtains the locale (Linux, UNIX) or code page (Windows) that is set on the client computer and performs a lookup of the corresponding codeset name in the ccsid.csv table to find the corresponding CCSID value to use. For example, if the code page for a Windows client is set to 1252, StarSQL looks for a codeset name of “WINDOWS-1252” in the ccsid.csv file.

The OverrideCodeset parameter can be useful if you want to send data encoded using a different codeset, or to match a local codeset name that is not in the ccsid.csv file to one that is. The value for the OverrideCodeset parameter can be any string.

### PackageFile

If you want to enable matching or recording for static SQL, set the package filename. The package file is used to bind static SQL packages on the DB2 host.

If recording is enabled, StarSQL records static SQL, and optionally DDL statements, to the package file. To enable recording, set Capture ([page 103](#)), CaptureDDL ([page 103](#)), or CaptureQryOnly ([page 103](#)).

If a valid package file name is entered and matching is enabled, StarSQL matches SQL statements in the application with SQL statements in the static SQL package on the host and executes those statements when it finds a match. To enable matching, set `UseStaticMatch` (see [page 114](#)).

## ParameterMarkersOnly

Set `ParameterMarkersOnly=Yes` to record only static SQL statements that have parameter markers to a package file. Set `ParameterMarkersOnly` to `No` if you want to record all SQL statements.

If you set `ParameterMarkersOnly=Yes`, you also must specify a value for the package file (see `PackageFile`, [page 110](#)) and enable one of the `Capture` settings (see [page 103](#)).

## PasswordProc

If you are using a stored procedure for password management, set `PasswordProc` to the name of the stored procedure. StarSQL includes a stored procedure named `DRDA11DE` to support password management on OS/400.

## PkgOwnID

The SQL Package Owner ID (`PkgOwnID`) specifies the owner ID to associate with the StarSQL driver packages that reside on the host. The target DBMS host uses the `PkgOwnID` value to validate whether the user has authority to perform the functions represented by the SQL statements in the bound package. This option can be useful in situations where an administrator needs to bind the driver packages but does not want to be the owner of the packages.

The default package owner is the user ID that is used to establish the ODBC connection when a package is bound. Enter up to 8 characters for the `PkgOwnID` option to specify a different user ID as the package owner. After you set the `PkgOwnID` option, run `StarAdmin` to bind the desired StarSQL packages with the package owner that was specified for `PkgOwnID`.

If you are using static SQL with the StarScribe Package Editor, note that specifying a `PkgOwnID` in the StarSQL driver configuration does not override the owner ID that is specified by StarScribe when binding custom SQL packages.

## QryBufSiz

Set the size of the query buffer. The buffer size can be between 512 bytes and 32767 bytes. The default is 32767 (32KB). A large query buffer can improve performance for large queries.

## ReadOnly

Set to Yes if the data source is read only. This prevents users from attempting to update the host database. Set to No if you want the StarSQL user to be able to update the host database (to the extent allowed by the host security).

## SpecialColumns

The SQLSpecialColumns function retrieves the following information about columns in a specific table:

- the optimal set of columns that uniquely identify a row
- columns that are automatically updated when any value in the row is updated

By default the SpecialColumns function is enabled (set to Yes). To disable the function, set the SpecialColumns keyword to No, which returns an empty results set if the SpecialColumns function is invoked.

## StrictParsing

When StrictParsing is enabled, which is the default, the SQL sent by an application must be acceptable to the StarSQL parser. If the SQL does not comply, StarSQL generates an error message and does not send the SQL.

If StrictParsing is disabled, StarSQL passes the SQL to the host database as-is. Some functionality, such as SQLDescribeParam(), may not be available to the application if StrictParsing is disabled.

## TypDefOvr

Use the TypDefOvr keyword to explicitly set the SBCS CCSID that StarSQL uses to send character data to the DB2 host. When TypDefOvr is set, StarSQL sends the data using the CCSID specified in the TypDefOvr with each DRDA request.

The CCSID specified for TypDefOvr overrides the CCSID values specified for AutoTypDefOvr (see [page 94](#)). The TypDefOvr keyword value can be any SBCS CCSID that StarSQL supports. (For the complete list of supported CCSIDs, refer to the “StarSQL Character Conversion and National Language Support” document in the StarSQL technical documents of the StarQuest Web site.)

If the CCSID specified for TypDefOvr is for a single-byte character set, StarSQL sends all SQL statements and parameters only as single-byte character strings defined by that CCSID. The section "[Connecting an MBCS Client to an SBCS Host](#)" on [page 125](#) provides additional information about customizing StarSQL to support specific client and host configurations.



## UseDSCRDBTBL

If the UseDSCRDBTBL setting is Yes, the default, StarSQL uses the DRDA DSCRDBTBL command to implement the SQLColumns ODBC function.

For hosts that do not support the DRDA DSCRDBTBL command, such as SQL/DS, DB2/UDB, or any host in a double-byte environment, set UseDSCRDBTBL to No. In this case, StarSQL implements SQLColumns by preparing and executing a SELECT command.

## UseDynamicCatalogSQL

Set the UseDynamicCatalogSQL to Yes if you want the StarSQL driver to use dynamic rather than static SQL for catalog functions when connected to DB2/zOS servers. The default value is Yes, causing dynamic SQL to be used when it's available.

## UseEncryption

The UseEncryption setting determines whether the login user ID and/or password is sent to the host encrypted or in clear text. When a user logs into the host or changes their host password using a StarSQL connection, StarSQL sends the user ID and password in clear text unless the UseEncryption keyword is set to Yes or Any. To send the user ID and/or password encrypted, you must set the UseEncryption keyword to Yes or Any and the host must support encryption.

DB2 for OS/390 v6.1 and z/OS v7 or later support or successfully negotiate password encryption. DB2 for i (OS/400) supports password encryption with the Cryptographic Access Provider described on [page 18](#), but does not support encrypted user IDs; therefore the UseEncryption setting must be Any or No. DB2 UDB supports encrypted passwords if you enable the database for encryption as described in "[Enabling Encryption](#)" on [page 58](#).

You can specify Any, Yes, and No for the UseEncryption setting, as described in the following table. The default value is Any.

**Table 18. UseEncryption Keyword Values**

UseEncryption Setting	Description
Any	StarSQL sends the user ID in clear text and the password encrypted. If the login fails, StarSQL then sends both the user ID and password in clear text.

Yes	StarSQL always sends the user ID and password to the host encrypted. If the host does not support encryption, StarSQL returns an error.
No	StarSQL always sends the user ID and password to the host in clear text.

## UseJumboPackages

The UseJumboPackages variable affects:

- how dynamic SQL packages are bound, and
- how many SQL statements handles are available when using the StarSQL driver to connect to a DB2 host.

If you configure the StarSQL DSN to use jumbo packages (UseJumboPackages=Yes), StarSQL sets the maximum active statement handle limit to 1,314. Each handle can each be used by one ODBC statement at a time. If you set UseJumboPackages to No, StarSQL binds the packages with a maximum of 64 statement handles.

To take full advantage of this option you must enable the UseJumboPackages variable before you bind StarSQL packages, and it must be enabled in the StarSQL data source that is used to connect to the database.

If a dynamic package that is bound with 64 sections (UseJumboPackages=No) receives 65 or more active statements because the StarSQL data source is configured to use jumbo packages, DB2 may report an error that the sections cannot be found. If packages are bound with the jumbo option enabled (UseJumboPackages=Yes), users can connect to DB2 using a StarSQL data source that is configured to use either the typical (64 statement handles) or jumbo-sized (1,314 statement handles) packages.

## UseStaticMatch

Set UseStaticMatch to Yes to enable matching of static SQL in the package file with SQL statements in the static SQL package on the host, and to execute those statements when StarSQL finds a match. You must specify a value for the PackageFile keyword (see "[PackageFile](#)" on page 110) if you set UseStaticMatch to Yes. The default for UseStaticMatch is No.

## UseSYSDUMMYAEU

The UseSYSDUMMYAEU keyword affects how LOB data is processed on a DB2 for z/OS host. The default setting of Yes indicates that the DB2 for z/OS system catalog contains the system tables SYSDUMMYA, SYSDUMMYE, and SYSDUMMYU, and StarSQL will use these tables instead of the standard SYSDUMMY1 table to avoid unnecessary conversion of LOB data that has a CCSID associated with it.

The SYSDUMMYA, SYSDUMMYE, and SYSDUMMYU tables are created by applying PTFs that IBM made available in APAR PQ85495. You can verify whether these tables exist on the host by entering the following statement:

```
SELECT * FROM SYSIBM.SYSTABLES WHERE NAME LIKE  
'SYSDUMMY%'
```

Note that only the SYSDUMMYA and SYSDUMMYE tables are created when applying the PTF to DB2 for z/OS v6—the SYSDUMMYU table is not created for that version. If your system has only the standard SYSDUMMY1 table and your database contains LOB data, the following error may be returned at runtime when users try to access LOB data:

```
SQLSTATE 42704 (-204) *[StarSQL] [StarSQL ODBC Driver] [DB2] IS  
AN UNDEFINED NAME*
```

To resolve this error you can:

- set the UseSYSDUMMYAEU keyword to No, or
- apply the PTFs from IBM APAR PQ85495 (<http://www-1.ibm.com/support/docview.wss?uid=swg1PQ85495>) to create the new SYSDUMMY tables.

If you set the UseSYSDUMMYAEU keyword to No, any double-byte EBCDIC or ASCII GRAPHIC data will be converted to UNICODE on the host before it is sent to StarSQL. If host performance is an issue, apply the PTFs and set UseSYSDUMMYAEU to Yes to avoid the unnecessary conversion.



# StarLicense Client Configuration Utility

---

You can use the StarLicense Client Configuration utility to manage which StarLicense server or servers are used to check out a license. The StarLicense Client Configuration utility allows you to:

- test that a license can be checked out
- configure which license server to use for connections from the local computer to a database
- remove a license server from the local configuration

Follow the steps below to start the StarLicense Client Configuration utility.

1. Log on to the computer as `root` user.
2. Change to the `$STARDIR/bin` directory where the StarLicense Client Configuration utility is installed.
3. Enter the following command to execute the startup script and display the StarLicense Client Configuration Menu.

```
# ./config-lic
```

The StarLicense Client Configuration Menu appears.

**Figure 4. StarLicense Client Configuration Menu**



## Adding a Connection to a StarLicense Server

Option 1 of the StarLicense Client Configuration Menu lets you define a connection to a StarLicense server. The StarLicense Server software can be installed on a remote computer or the local computer. After you select Option 1 a screen appears so you can specify information about the StarLicense server you want the client to use for licensing.

1. Enter the hostname or TCP/IP address of the StarLicense server. (If the StarLicense Server software is installed on the local computer you can specify the loopback address 127.0.0.1 or "localhost".)
2. Enter the port number that the server is using to listen for license requests. The default port number is 4999.
3. Enter the product ID for the license.

After you provide the StarLicense server information the StarLicense Client Configuration utility shows the client-server connection being added.

## Removing a StarLicense Server Definition

Option 2 of the StarLicense Client Configuration Menu lets you remove a StarLicense server from the local computer's configuration. After you select Option 2 a screen appears with information similar to the following:

```
Hostname=127.0.0.1 Port=4999 ProductID=SQ PRIMARY
Delete this connection (y/n)?
```

If there is more than one connection defined, respond with N until the connection that you want to remove appears and then enter Y to confirm you want to remove the connection.

## **Testing License Checkout**

Option 3 of the StarLicense Configuration Menu lets you test that one or more licenses can be checked out. After you configure a license server it is good practice to execute this option to verify that licenses can be checked out.

After you select Option 3 a prompt appears so you can enter the number of licenses to check out. The default is 1 license, but you can enter any number. After you specify the number of licenses, the utility attempts to check out that number of licenses for the particular product ID and displays the results. Press Enter to check the license(s) back in so they will be available for other connections.

## **Displaying the StarLicense Configuration**

Option 4 of the StarLicense Client Configuration Menu displays information about the current configuration of StarLicense. The information is categorized by Client License Servers, Client License Keys, Server License Keys, and Primary Server Entries, some of which may not be applicable to StarSQL for UNIX licensing. Configuration details include:

- the license keys that have been configured
- which StarQuest product the key is for
- when the license expires
- how many connections the license allows
- the IP address and port that listeners are configured to use
- the primary and secondary servers if more than one remote StarLicense server has been configured
- the logging level configured for the server
- the journaling level configured for the server
- the number of days that the server journal entries are retained

## Setting the StarLicense Client Logging Level

Option 5 of the StarLicense Client Configuration Menu allows you to enable logging of the client license activity. Logging information may be needed by StarQuest Customer Support to help troubleshoot a problem with client licensing. If licensing is working fine there is no need to change the logging level from the default value of 0 (zero).

The valid values for the logging level are:

0	no logging
FFFFFFFF	log all license activity

A new log file is created for each process and written to the `/tmp` directory. The filename format for the log file is:

```
starliccli.<processID>.log
```

The log file contains the timestamp, a message class identifier, and a text string that describes the activity of the license service. Be sure to set the logging level back to 0 (zero) when you no longer need to capture license service activity to avoid creating unnecessary log files.



# StarSQL National Language Support

---

StarSQL is a v3.0-compliant ANSI ODBC driver that provides connectivity to any database that is compliant with the Distributed Relational Database Architecture (DRDA). This includes all of IBM's DB2 family of database products running on mainframe, midrange and workstation-based platforms. StarSQL allows connecting to DB2 from all modern Microsoft Windows and most popular UNIX- and Linux-based computers.

To support a wide range of host systems, StarSQL includes conversion support for character encoding based upon Unicode, IBM EBCDIC (including Extended EBCDIC) and ANSI (ASCII) standards. In addition, StarSQL supports legacy UNIX encoding, such as Extended UNIX Coding (EUC) and the International Standards Organization (ISO) encoding.

This appendix provides background of how characters are encoded and data is converted from one system to another, and how StarSQL supports using different languages and character encoding schemes. It is intended for any reader who is interested in character conversion, but is especially pertinent for StarSQL users who are:

- running a client that is configured with a code page for one of the Group 2 languages (Japanese, Chinese, or Korean) that needs to connect to a host that does not support these languages.
- accessing a DB2 z/OS system that is supporting one of the Group 2 languages using extended EBCDIC.

The section "[Customizing StarSQL for National Language Support](#)" on page 124 specifically addresses using StarSQL in the above situations.

## Character Encoding of Data Across Systems

Most legacy (pre-Unicode) client environments use the ASCII-derived character encoding that was developed by the ANSI standards body to store and manipulate character strings. ASCII encoding allowed each ASCII character to be stored as a byte, with the initial version of ASCII using only 7 of the 8 bits available in a byte. This allowed applications to use 128 different characters. ASCII was subsequently extended to support most European characters by using the eighth bit to expand the total range of characters to 256. Currently ASCII refers to either the 7-bit or 8-bit encoding of characters. The Group 2 languages require many more characters than can be encoded by a single byte. These languages typically employ a variety of extensions to 7-bit ASCII, using the high-order bit to designate the use of an additional byte to encode more characters (approximately 32K). Such encoding schemes are termed “multi-byte” character encoding and several distinct de jure standards exist, including ANSI and ISO, in addition to de facto standards such as EUC and IBM’s Extended EBCDIC.

Today most Windows systems provide support for ANSI or Unicode encoding schemes, and most UNIX systems provide support for ISO or Unicode. Most IBM host systems support EBCDIC and Unicode. StarSQL is designed to support connectivity among these different platforms.

To distinguish various encoding schemes, IBM created a standard enumeration of virtually all character encoding using a 16-bit value. This unique value is called the Coded Character Set Identifier (CCSID), and it is used to tag all character data exchanges between StarSQL and the host DBMS. The CCSID is just a number that identifies a particular code page. For example, a host in the U.S. or Canada would typically use the US-English code page denoted by a CCSID of 37, whereas a host in Germany may use a code page represented by CCSID 273.

Character data in most Linux, UNIX, and Windows environments is represented only by a code page, which StarSQL treats as a CCSID. For example, the U.S. form of English using ANSI encoding is CCSID 1252.

StarSQL uses the CCSID to identify client and host encoding schemes in a variety of ways, including for Expert Settings that modify the behavior of the driver.

## StarSQL and Unicode

StarSQL uses Unicode internally to parse SQL statements and as the basis for converting unlike data encoding between connected systems. This use of Unicode internally also allows StarSQL to efficiently receive Unicode from modern host systems that use Unicode as the native encoding scheme.

## Determining and Setting the Client Code Page

StarSQL determines the local client's code page by a variety of system calls—the value is normally returned as part of the “locale” API support.

On a Windows-based computer the system language is specified in the Regional and Language Options of the Control Panel. The code pages that determine how non-Unicode characters are converted are represented by a numeric string, such as 1252 for US-English code page. StarSQL converts the code page number to a string that the conversion routines can process, such as WINDOWS-1252.

On a Linux- or UNIX-based computer, you specify what code page to use by setting the locale, which is specified by the **LANG** environment variable. You can modify the locale using the **LANG**, **LC-CTYPE**, or **LC\_ALL** environment variables.

## Determining the Host Code Page

The code pages that are specified for the host to use are detected as part of the initial exchange between StarSQL and DB2. You can determine the CCSIDs that the host is returning to StarSQL by connecting to DB2 using a StarSQL data source. After you connect to the DB2 host, look at the AutoTypDefOvr parameter in the `.odbc.ini` file to see the CCSID values that the host returned to StarSQL. Use the view command, as shown below, or any text editor, to display the contents of the initialization file.

```
> view .odbc.ini
```

The AutoTypDefOvr setting is three comma-separated values of the CCSID for SBCS, DBCS, and MBCS, respectively. For example, the following AutoTypDefOvr value indicates that the DB2 host returned CCSID 37, with no CCSID value set for DBCS or MBCS.

```
AutoTypDefOvr=37,0,0
```

To control the SBCS CCSID that StarSQL uses to send outbound data, you can modify the TypDefOvr value of the StarSQL data source as described in ["Customizing StarSQL for National Language Support" on page 124](#).

## StarSQL Implementation Details

The default behavior of the StarSQL driver typically involves the following steps when connecting to a DB2 host and exchanging SQL data.

1. StarSQL connects to DB2 and obtains the CCSID values that the host expects for SBCS, DBCS, and MBCS data. These CCSID values are stored in the AutoTypDefOvr setting of the StarSQL data source for future use.

2. StarSQL sends SQL statements and input data to DB2, converting SBCS data to the host-specified CCSID for single-byte data. Mixed-byte data is sent using the client code page, and the host is responsible for converting it to the host-specified CCSID for DBCS and MBCS data.
3. DB2 converts the data to the host server's code page, if necessary, and then processes the data.
4. DB2 sends the result back to the StarSQL client.
5. StarSQL converts the result to the code page of the client computer.

StarSQL performs inbound data conversion from the host system based upon the mappings that are defined in the `ccsid.csv` table that is installed with StarSQL. The `ccsid.csv` table is platform-specific, and is installed to the `\Programs` directory of a Windows-based computer or to the `$STARDIR/etc/conf` subdirectory of a Linux- or UNIX-based computer.

In order for StarSQL to convert data from one character set to another, there must be a mapping defined in the `ccsid.csv` table for all the CCSIDs used in the exchange. If there is no mapping defined for a specified CCSID, an error is reported. Support for additional character conversions may be requested from StarQuest Customer Support, and you also may be able to customize the StarSQL data source to use a different CCSID that is supported as explained in the next section, “[Customizing StarSQL for National Language Support](#).”

If one or more characters cannot be converted between the host and client character sets, StarSQL substitutes a question mark, `?`, in place of the character(s). You can configure whether the substitution generates a warning by setting the `CharacterSubstitution` parameter of the StarSQL data source (see [page 104](#)).

## Customizing StarSQL for National Language Support

This section addresses situations that may require customizing StarSQL to support a specific client connecting to a specific DB2 host: StarSQL may need to be customized in order to:

- establish a connection between a client that is configured with a code page for one of the Group 2 languages (Japanese, Chinese, or Korean) and a host that does not support these languages.
- access a DB2 z/OS system that is supporting one of the Group 2 languages using extended EBCDIC.

## Connecting an MBCS Client to an SBCS Host

IBM encoding of the MBCS support recognizes two distinct CCSIDs for SBCS and DBCS encoded characters. Typically only characters encoded using the SBCS component are used in SQL statements, while both DBCS and MBCS data can be the output of the SQL statements. If a host is not configured to support mixed-byte character sets it can receive only SBCS data.

Typically StarSQL sends SBCS data to the host with characters encoded according to the CCSID value that the host indicates it is using for SBCS data when it returns the `AutoTypDefOvr` to StarSQL. You can specify a particular CCSID to use for converting SQL data by setting the `TypDefOvr` parameter of the data source that StarSQL uses to connect to the host.

The `TypDefOvr` setting overrides the SBCS CCSID value that is specified in the `AutoTypDefOvr` setting. When converting SQL data to a form the host can use, StarSQL will use the conversion routine that is associated with the CCSID in the `TypDefOvr` setting, regardless of the client computer's character encoding scheme (SBCS or MBCS) or the `AutoTypDefOvr` values.

If a host reports to StarSQL that a CCSID being used by the client is not acceptable, useful connectivity may still be achieved by specifying an SBCS CCSID that is supported by the host, which restricts the SQL character data to only single-byte character strings. For example, when connecting a client computer that is using a Japanese character set to a U.S.-based host that does not support Japanese mixed-byte character set, you can force a specific SBCS CCSID to be used by setting the `TypDefOvr` setting of the StarSQL data source.

## Supporting Asian Languages with Extended EBCDIC

Some multi-byte host environments have restrictions regarding the type of SQL phrases which may be used. Specifically, hosts that are running DB2 for z/OS with support for Group-2 languages may not handle the default SQL statements used by StarSQL to fulfill ODBC catalog calls.

When using Extended EBCDIC client encoding, StarSQL packages must be bound with the `CustomizePrdid` parameter enabled for "Mixed" (see [page 105](#)). Setting the `CustomizPrdid` parameter to M (Mixed) disables use of the backslash character, `\`, as an escape character in arguments passed to catalog functions. The escape character precedes a special character, such as the underscore (`_`) character, to treat the character as a literal. To work with tables that include an underscore in the name, such as `MY_TABLE`, create an `ALIAS`, `SYNONYM`, or `VIEW` for the table with a name that includes no special characters instead of working directly with the table.

StarSQL packages are normally bound using the StarAdmin utility that is included with the Windows version of StarSQL. Be sure that the DSN has CustomizePrdid set to Mixed before binding packages with the StarAdmin utility. The ODBC DSN that StarSQL clients use to connect to the host also must be configured with CustomizePrdid set to Mixed to use the packages that are bound with CustomizePrdid set to Mixed.

## **Currently Supported CCSIDs**

StarSQL supports converting character data between a wide range of CCSID-to-CCSID pairs and CCSID-to-code-page pairs. It supports Group 1, Group 1A, and Group 2 character sets as defined by CDRA.

- Group 1 covers the Roman Alphabet Number 1, which includes Australia, Hong Kong, New Zealand, North and South America, and Western Europe.
- Group 1A covers multilingual scripts Cyrillic, Hebrew, Greek, and Turkish. The Latin 2 character set associated with Central Europe is supported in this group.
- Group 2 covers double-byte coding for Japan, Korea, the People's Republic of China, the Republic of China, and Thailand.

For the complete list of supported CCSIDs, refer to the “StarSQL Character Conversion and National Language Support” document in the StarSQL technical documents of the StarQuest Web site.

# Glossary

---

**APAR - Authorized Program Analysis Report** A request for correction of a problem caused by a defect in a current unaltered release of a program.

## **APPC**

APPC (Advance Program-to-Program Communications) (1) The general facility characterizing the LU 6.2 architecture and its various implementations in products. (2) Sometimes used to refer to the LU 6.2 architecture and its product implementations as a whole, or to an LU 6.2 product feature in particular, such as an APPC application programming interface.

**APPL** An APPL statement defines the DB2 subsystem to VTAM for the purposes of remote access. It is required for any configuration that involves a DB2 host on an IBM mainframe (MVS or OS/390).

**authorization identifier** On DB2 for z/OS and DB2 for VM and VSE, the authorization identifier is user's login ID or an assigned Authorization Identifier, which corresponds to a group with which the user is associated.

**BSDS** The BSDS (bootstrap data set) is a VSAM data set that contains name and status information for DB2, as well as RBA range specifications, for all active and archive log data sets, passwords for the DB2 directory and catalog, and lists of conditional restart and checkpoint records.

**CCSID** CCSID (Coded Character Set Identifier) represents a character set, code page, encoding scheme, and additional coding-related information. Used to support international code sets.

**collection** A collection is a location on the host for database objects, such as user tables and catalog tables, which are collected together under a single qualifying name.

- CoS** Class of Service (CoS) is the ability of switches and routers to prioritize traffic into different queues and classes.
- data source** A data source describes a connection to a database from an ODBC application.
- DDF** DDF (Distributed Data Facility) is the vehicle that DB2 uses to send and receive remote procedure calls.
- DRDA** DRDA (Distributed Relational database Architecture) supports access to distributed data by which an application can explicitly connect to another location, using an SQL statement, to execute packages that have been previously bound at that location.
- DSN** A DSN (Data Source Name) is an ODBC definition that refers to a particular database.
- Dynamic SQL** Dynamic SQL refers to SQL statements that are prepared and executed within an application program while the program is executing. A dynamic SQL statement can change during program execution. Contrasted with Static SQL.
- held cursor** A held cursor is a cursor that is not automatically closed when its transaction commits.
- installable image** An installable image is an image of a StarSQL installation that can be installed on multiple desktops over the network.
- isolation level** The isolation level refers to the degree of concurrency permitted in a transaction.
- logmode** Same as mode.
- LU** An LU (Logical Unit) is a component of an SNA connection. Each end of the connection requires a unique LU name.
- LU6.2** A type of logical unit that supports general communication between programs in a distributed processing environment. LU 6.2 is characterized by (a) a peer relationship between session partners, (b) efficient utilization of a session for multiple transactions, (c) comprehensive end-to-end error processing, and (d) a generic application programming interface (API) consisting of structured verbs that are mapped into a product implementation.
- mode** The mode contains SNA session-specific information, such as the packet size (RU Size) that must be synchronized before an SNA session can be initiated.



- ODBC** ODBC (Open Database Connectivity) is a call-level interface developed by Microsoft Corporation that allows a single application to access DBMSs from different vendors using a single interface.
- Open Edition** Open Edition is a component of the MVS operating system that supports TCP/IP processing for certain levels of OS/390.
- package** A package is an object on the host containing a set of SQL statements that have been bound statically and are available for processing.
- PTF** Program Temporary Fix - a method used by IBM for distributing fixes quickly.
- PU** A PU (Physical Unit) is a component of an SNA session. A PU requires certain characteristics to support LU6.2 sessions.
- RDB Name** An RDB name is a unique identifier for an RDBMS within a network.
- REGEDIT** The registry editor supplied by the Windows operating system for manually editing the Registry.
- RU** An RU (Request Unit) is the portion of a basic information unit that follows a request header and contains the data. (SNA)
- Side Information Record** The Side Information Profile is a component of an SNA connection that includes session-specific information, such as local and remote program names, session mode name, security type, user ID, and password. This is used by CIP-C.
- SNA** SNA (Systems Network Architecture) is a network protocol used extensively by IBM products. See also APPC.
- Static SQL** Static SQL refers to SQL statements in an application program that are created before the application executes. After being created, a static SQL statement does not change, although values of host variables specified by the statement might change. Contrasted with Dynamic SQL.
- system catalogs** The system catalogs are tables in the host database that DB2 uses to keep track of the system. On some RDBMSs, they are called the system tables.
- TCP/IP** TCP/IP (Transmission Control Protocol / Internet Protocol) is a commonly - used network protocol.
- TP name** The TP Name (APPC Transaction Program Name) defines the name of the remote transaction program that the database client must use when it issues an allocate request to the database server when using the APPC communication protocol.
- user id** A user id is a unique identifier that enables a user to logon to a host.

**VTAM** VTAM (Virtual Telecommunications Access Method) provides network communications on IBM mainframes and AS/400 systems.

**VSAM** Virtual Storage Access Method - A data storage system used in IBM machines. VSAM was designed to organize data more efficiently and to improve access time by searching indexes instead of actual files.

**VM** Virtual Machine - An IBM virtual data processing system in which multiple operating systems and program can be run by the computer at the same time. Each user appears to have an independent computer with its own input and output services.

**VSE** Virtual Storage Extended - A system that consists of a basic operating system (VSE/Advanced Functions) and any IBM supplied and user-written programs required to meet the data processing needs of a user. VSE and the hardware it controls form a complete computing system. Its current version is called VSE/ESA.

# Index

---

\$STARDIR 16  
? substitution character 104, 124

## Symbols

\_ character, limitations for 105, 125

## A

access rights 94, 112  
Accounting keyword 99  
accounts, user 46  
AIX  
    installing StarSQL on 71  
    removing StarSQL from 26  
    versions supported 23, 27  
AllowSynonyms keyword 100  
AlwUpd keyword 94  
applications, developing StarSQL 42  
AS/400 host 51  
ASCII 121  
Asian language support 125  
AutoBind keyword 68, 100  
AutoSqlSet keyword 101  
AutoTypDefOvr keyword 94, 101, 123

## B

binary data 102  
BinaryCCSID keyword 102  
bind option, EXPLAIN 95  
binding packages 62, 63, 66, 100, 110

BindRules keyword 102  
BLOB data type 13  
buffer  
    command 104  
    query 111

## C

CacheUID keyword 95, 103  
Capture keyword 103  
CaptureDDL keyword 103  
CaptureQryOnly keyword 103  
capturing explain information 62, 95  
catalog functions  
    filtering 103  
    query results 14  
    system tables for 104  
catalog package 63  
CatFilters keyword 103  
CatQual keyword 104  
CCSID  
    binary data 102  
    conversion table 124  
    for national language support 121  
    in AutoTypDefOvr 94, 101  
    in TypDefOvr 112  
    list of currently supported 126  
    overriding 125  
    values, host 123

## Index

- ccsid.csv table 124
- character encoding 121, 122
- character sets, international 121
- character substitution 124
- CharacterSubstitution keyword 104, 124
- charge-back accounting 99
- checkout, testing license 119
- client code page 123
- client connection information 101
- CLOB data type 13
- CmdBufSize keyword 104
- code page
  - client 123
  - host 123
- Coded Character Set Identifiers
  - see CCSID
- codeset name, overriding the 110
- collection of packages 39, 96
- command(s)
  - config-lic 82
  - CRTLIB 51
  - installp 72
  - pkgviewer 82
  - restart 84
  - simpleconn.odbc 40, 86
  - starlic-clientcfg 88
  - starping 40, 90
  - trcstart 91
  - trcviewer 92
  - WRDSRVTBLE 53
  - WRKTCPSTS 53
- compatibility
  - for LOBs, backwards 109
  - package 62
- concurrency, controlling 108
- config-lic command 82
- configuration files 93, 119
- configuration keywords
  - Accounting 99
  - AllowSynonyms 100
  - AlwUpd 94
  - AutoBind 68, 100
  - AutoSqlSet 101
  - AutoTypDefOvr 94, 101
  - BinaryCCSID 102
  - BindRules 102
  - CacheUID 95, 103
  - Capture 103
  - CaptureDDL 103
  - CaptureQryOnly 103
  - CatFilters 103
  - CatQual 104
  - CharacterSubstitution 104
  - CmdBufSize 104
  - CreateTable 104
  - CustomizePrdid 105
  - DefaultQualifier 106
  - Description 106
  - Driver 96
  - ExpirationWarning 106
  - Explain 95
  - FetchAhead 106
  - FoldUIDPWD 106
  - GetInfoCatalogUsage 106
  - HeldCursors 107
  - HostName 39, 96
  - IncludeSynonyms 108
  - IsolationLevel 108
  - KeepDynamic 109
  - LiteralConversion 109
  - LongStrParams 109
  - MaxRows 110
  - OverrideCodeset 110
  - PackageFile 110
  - ParameterMarkersOnly 111
  - PasswordProc 111
  - PkgColID 39, 96
  - PkgOwnID 111
  - Port 39, 97
  - QryBufSiz 111
  - ReadOnly 112
  - Server 39, 97
  - SpecialColumns 112
  - StrictParsing 112

- TypDefOvr 112
  - UseDSCRDBTBL 113
  - useDynamicCatalogSQL 113
  - UseEncryption 113
  - UseJumboPackages 114
  - UseStaticMatch 114
  - UseSYSDUMMYAEU 115
  - configuring
    - data sources 36, 45, 93
    - licenses 117
    - National Language Support 124
    - required DSN parameters 39, 96
    - StarLicense 117
    - unixODBC driver manager 34
  - connecting to StarLicense server 118
  - connection
    - security for 95, 103
    - testing the StarSQL/DB2 40
  - contacting StarQuest 20, 21
  - conversion table for CCSIDs 124
  - conversion tables 121
  - conversion, character 104
  - CreateTable keyword 104
  - CRTLIB command 51
  - CustomizePrdid keyword 105, 125
- D**
- data formats 121
  - data source(s)
    - access rights to 112
    - configuration parameters 39, 96, 97
    - configuring 36, 93
    - description 106
    - file location 36
    - global data settings 94, 99
    - host information for 45
    - name 36
  - data types 13
  - data, converting 121
  - database
    - access rights to 112
    - name, determining 52
    - name, specifying 39, 97
    - names, listing 59
    - server name 39, 96, 97
  - DB2 host(s)
    - DRDA port on 57
    - enabling DRDA on a 53
    - on MVS 47
    - preparing an AS/400 51
    - RDB name on 52
    - supported 12
    - system tables 104
    - testing connection to 40
  - DBCLOB data type 13
  - DDF record
    - starting 48
  - DDL statements, recording 103
  - DefaultQualifier keyword 106
  - Description keyword 106
  - description, data source 106
  - directive, table 104
  - documentation, list of IBM 17
  - DRDA 53
    - standard 121
    - tracing 91, 92
  - Driver keyword 96
  - driver manager, unixODBC 34
  - driver pathname 96
  - DSCRDBTBL command 113
  - DSN
    - configuration 36, 97
    - configuration file 95
    - definition 36
  - DSNJU003 utility 47
  - DSNTIPR panel 47
  - dynamic SQL 61, 63, 102
  - dynamic SQL, using 113
- E**
- Easy Install method 25, 29
  - EBCDIC 121, 125
  - enabling DRDA 53
  - encoding, character 121

## Index

encrypting logins 113  
encryption, password 13  
environment variable  
    LANG 123  
expiration, password 106  
ExpirationWarning keyword 106  
Explain facility 62, 95  
Extended UNIX Coding (EUC) 121

## F

features, new 9  
FetchAhead keyword 106  
file(s)  
    data source configuration 93  
    .odbc.ini 93, 95  
    package 82, 110  
    path to driver 96  
    .swodbc.ini 93, 95

FoldUIDPWD keyword 106

## FreeBSD

    installing StarSQL on 73  
    removing StarSQL from 26  
    versions supported 23, 27

## G

GetInfoCatalogUsage keyword 106  
global configuration file 95  
global data source settings 94, 99  
Group-2 languages 122, 125

## H

header files, StarSQL 42  
HeldCursors keyword 107  
host(s)  
    DB2 for i 51, 53  
    determining the code page for 123  
    information for data sources 45  
    MVS 47  
    passwords, changing 49  
    permissions to 47  
    preparation of 45  
    supported DB2 12

    UDB 57

    user accounts 46

HostName keyword 39, 96

## HP-UX

    installing StarSQL on 73  
    removing StarSQL from 26  
    versions supported 23, 27

## I

IBM documentation 17  
IDs, user 46  
IncludeSynonyms keyword 108  
installing  
    StarSQL 25, 28  
    stored procedures 53  
installp command 72  
international language support 121  
International Standards Organization (ISO)  
    encoding 121  
IP port, setting the 57  
IsolationLevel keyword 108

## K

KeepDynamic keyword 109  
key, obtaining a license 20

## L

LANG environment variable 123  
language support, international 121  
level, concurrency 108  
library, package 51  
license  
    checkout, testing 119  
    configuring 117  
    configuring a 82  
    using a server 32  
license key, obtaining 20  
licensing StarSQL 88  
linked servers, sending queries to a 106  
Linux  
    installing StarSQL on 75, 79  
    removing StarSQL from 26, 29

- system requirements 23, 27
- versions supported 23, 27
- listing database names 59
- LiteralConversion keyword 109
- literals, substituting 109
- LOB data types 13, 62, 109
- LOB data, processing 115
- local code page 123
- locale 123
- location name 39, 97
- logins, encrypting 113
- LongStrParams keyword 14, 109

## M

- management, password 49, 53, 106
- manager, driver 34
- matching static SQL 84, 109, 110, 114
- MaxRows keyword 110
- mixed-byte character sets 125
- mixed-byte client computers 125
- MVS host
  - charge-back accounting for 99
  - preparing DB2 on an 47

## N

- name, overriding the codeset 110
- names, using catalog and schema 106
- national languages, support for 121
- network
  - connection, testing the 40
  - requirements 12
- new features 9

## O

- ODBC
  - catalog functions 61, 103, 104
  - compliance 121
  - SQLColumns function 113
  - SQLSpecialColumns function 112
- ODBCConfig utility 10
- .odbc.ini file 93, 95
- OS/390, charge-back accounting for 99

- OverrideCodeset keyword 110
- overriding the CCSID 125

## P

- package owner ID 111
- package(s) 61
  - automatic binding of 68, 100
  - binding 62
  - binding with Explain 62, 95
  - catalog 63
  - collection ID 39, 96
  - compatibility 62
  - configuring
    - a license 82
  - dynamic SQL 63
  - libraries for 51
  - permissions for 47, 66
  - recording DDL statements to 103
  - recording SELECT statements to 103
  - recording static SQL to 103
  - static SQL 66
  - viewing 82
- PackageFile keyword 110
- page, code 123
- ParameterMarkersOnly keyword 111
- parameters markers, recording 111
- parsing SQL statements 112
- password encryption 13
- password management 49, 53, 106, 111
- password, encrypting 113
- password, folding to uppercase 106
- PasswordProc keyword 111
- passwords, user 46
- pathname, driver 96
- performance, improving 104, 106, 111
- permissions for packages 47, 66
- PkgCollID keyword 39, 96
- PkgOwnID keyword 111
- pkgviewer command 82
- port
  - for DRDA requests 53
  - for TCP/IP connections 39, 57, 97

## Index

- Port keyword 39, 97
- prepared SQL statements 109
- preparing
  - a DB2 for i host 51
  - a DB2/UDB host 57
  - DB2 on an MVS host 47
  - hosts for StarSQL 45
- Q**
- QryBufSiz keyword 111
- qualifying objects 106
- qualifying SQL statements 106
- query buffer 111
- R**
- RDB name 52
- ReadOnly keyword 112
- recording static SQL 84, 110
- restart command 84
- requirements
  - host 12
  - network 12
  - worstation 23, 27
- results set
  - empty 112
  - limiting 110
- S**
- security, connection 95, 103
- SELECT statements, recording 103
- Server keyword 39, 97
- server, connecting to StarLicense 118
- silent character substitution 104
- simpleconn.odbc command 40, 86
- SMIT, using to install StarSQL 71
- Solaris
  - installing StarSQL on 77
  - removing StarSQL from 26
  - versions supported 23, 27
- SpecialColumns keyword 112
- SQL Server
  - linked servers, using 106
- SQL statement(s)
  - dynamic 63
  - improving performance of 104, 111
  - matching 114
  - parsing 112
  - prepared 109
  - qualifying 106
  - recording 111
  - static 66
  - table directive 104
- SQLColumns function 113
- SQLGetInfo() 106
- SQLSpecialColumns function, enabling 112
- StarAdmin utility 10
- starlic-clientcfg command 88
- StarLicense
  - configuration 119
  - Configuration Menu 117
  - server 118
- StarLicense server 32
- starping command 40, 90
- StarQuest, contacting 20, 21
- StarScribe Package Editor 66, 82
- StarSQL
  - and Unicode 122
  - applications, developing 42
  - data exchange 123
  - Easy Install method 25, 29
  - header files 42
  - installing 25, 28
  - licensing 88
  - manually installing 71, 79
  - national language support 121
  - overview 9
  - preparing hosts for 45
  - removing 26, 29
  - system requirements 23
  - testing 40, 86, 90
  - upgrading 24, 26
  - versions comparison 9, 24, 26
- starting DDF 48
- statement handle limit 114



static SQL 61, 66, 109, 110  
 stored procedure, password management 53, 111  
 StrictParsing keyword 112  
 strings, mapping of DB2 14  
 .swodbc.ini file 93, 95  
 synonyms, retrieving 100  
 system data sources 36  
 system requirements 23

## T

tables, creating 104  
 TCP/IP  
   connections, port for 39, 97  
   enabling DRDA over 53  
   setting the IP port 57  
   UDB host address for 57  
 technical support, StarQuest 20  
 testing licenses 119  
 testing StarSQL 40, 86, 90  
 tracing  
   DRDA 91, 92  
   unixODBC driver manager 35  
 transactions  
   access rights to 94  
   controlling concurrency of 108  
 trcstart command 91  
 trcviewer command 92  
 TypDefOvr keyword 112, 125

## U

UDB host 57  
 underscore character, limitations for 125

Unicode 122  
 unixODBC driver manager 10, 34  
   configuring the 34  
   testing the 40, 86  
   tracing 35  
 upgrading StarSQL 24, 26  
 uppercase, folding UserID and Password to 106  
 UseDSCRDBTBL keyword 113  
 UseDynamicCatalogSQL keyword 113  
 UseEncryption keyword 113  
 UseJumboPackages keyword 114  
 user  
   accounts 46  
 user data sources 36  
 UserID, folding to uppercase 106  
 UseStaticMatch keyword 114  
 UseSYSDUMMYAEU keyword 115  
 utility  
   odbcAdmin 10  
   ODBCConfig 36  
   StarAdmin 10, 62

## V

verifying license checkout 119  
 versions, StarSQL 9  
 viewing a package file 82

## W

workstation requirements 23, 27  
 WRDSRVTBLE command 53  
 WRKTCPSTS command 53

