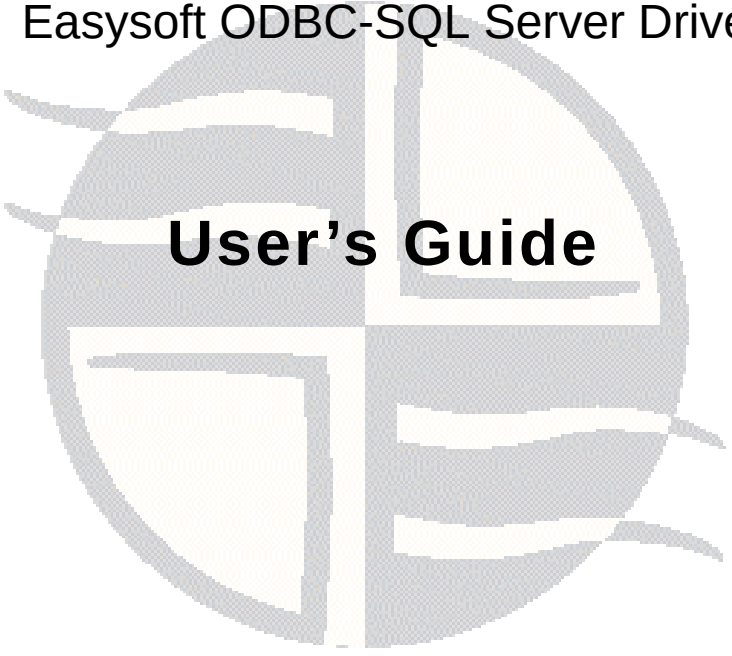
Easysoft Data Access

Easysoft ODBC-SQL Server Driver



User's Guide

This manual documents version 1.0 of the Easysoft ODBC-SQL Server Driver.

Publisher: Easysoft Limited

Thorp Arch Grange

Thorp Arch

Wetherby

LS23 7BA

United Kingdom

CONTENTS

List of Figures **5**

Chapter 1 Preface **7**

Intended Audience 8
Displaying the Manual 8
Notational Conventions 9
Typographical Conventions 10
Contents 11
Trademarks 12

Chapter 2 Introduction **13**

Product Status 14
Deployment 14

Chapter 3 Installation **15**

Obtaining the Easysoft ODBC-SQL Server Driver 16
What to Install 17
Installing the Easysoft ODBC-SQL Server Driver 19
Uninstalling the Easysoft ODBC-SQL Server Driver 48

Chapter 4 Configuration 51

Testing the Connection to SQL Server with tdshelper 52
Configuring the Easysoft ODBC-SQL Server Driver 60
Setting Up Data Sources 61
DSN-less Connections 81

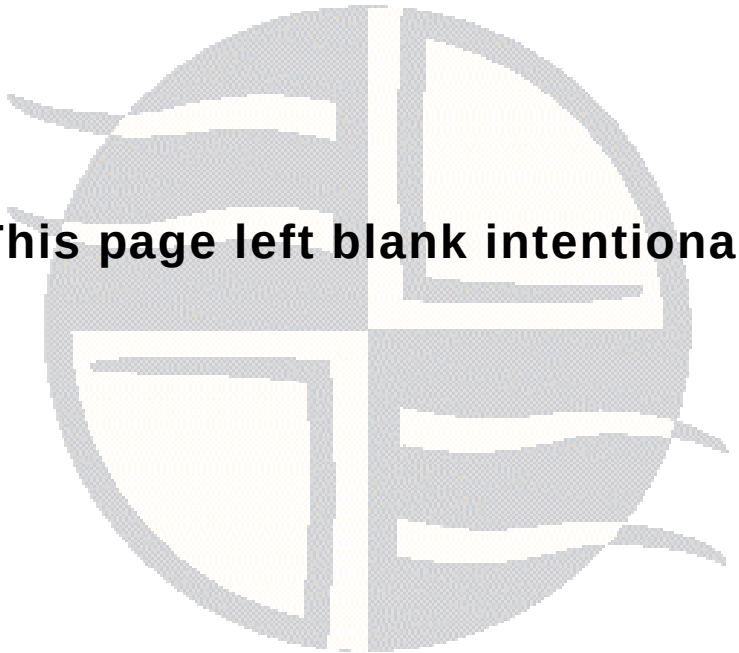
Chapter 5 Technical Reference 83

Restrictions 84
ODBC Conformance 84
Unicode Support 87
The SQL Server 2005 xml Data Type 88
Using Large-Value Data Types 91
SQL Server Authentication Modes 92
Encrypting Connections to SQL Server 94
Threading 107
Tracing 108

Chapter 6 Glossary 111

LIST OF FIGURES

Figure 1: Easysoft unixODBC configure line options.. 33
Figure 2: Dynamic linker search path environment variables. 46
Figure 3: Easysoft ODBC-SQL Server Driver data source settings.. 72
Figure 4: Easysoft ODBC-SQL Server Driver with SSL Support data source settings. 106



This page left blank intentionally

PREFACE

About this manual

This manual is intended for use by anyone who wants to install the Easysoft ODBC-SQL Server Driver, configure it, and then access SQL Server data sources from an ODBC-enabled application.

Chapter Guide

- **Intended Audience**
- **Notational Conventions**
- **Typographical Conventions**
- **Contents**
- **Trademarks**

Intended Audience

The Unix-based sections require experience of using Unix shell commands. You need to be able to do basic tasks such as editing text files. More complex tasks are described in detail, but it helps to understand how your system handles dynamic linking of shared objects.

Displaying the Manual

This manual is available in the following formats:

- Portable Document Format (PDF), which can be displayed and printed by using the Adobe Reader, available free from Adobe at **http://www.adobe.com**.
- HTML.

Notational Conventions

A *note box* provides additional information that may further your understanding of a particular topic in this manual:

Note	Note boxes often highlight information that you may need to be aware of when using a particular feature.

A *platform note* provides platform-specific information for a particular procedural step:

Linux	On Linux, you must log on as the `root` user to make many important changes.

A *caution box* provides important information that you should check and understand, prior to starting a particular procedure or reading a particular section of this manual:

Caution!	Be sure to pay attention to these paragraphs because Caution boxes are important!

Typographical Conventions

This manual uses the following typographical conventions:

- User interface components such as icon names, menu names, buttons and selections are displayed in **bold**, for example:

  Click **Next** to continue.

- Commands to be typed are displayed in a `monotype` font, for example:

  At the command prompt, type `admin`.

- File listings and system names (such as file names, directories and database fields) are displayed in a `monotype` font.

Contents

- **Introduction**

  Introduces the Easysoft ODBC-SQL Server Driver.

- **Installation**

  Explains how to install the Easysoft ODBC-SQL Server Driver.

- **Configuration**

  Explains how to configure the Easysoft ODBC-SQL Server Driver.

- Appendices

  Technical Reference and **Glossary**.

Trademarks

Throughout this manual, *Windows* refers generically to Microsoft Windows 95, 98, 2000, NT, XP, ME or 2003 Server, which are trademarks of the Microsoft Corporation. The X Window system is specifically excluded from this and is referred to as *The X Window System* or just *X*.

Note also that although the name UNIX is a registered trademark of The Open Group, the term has come to encompass a whole range of UNIX-like operating systems, including the free, public Linux and even the proprietary Solaris. Easysoft use Unix (note the case) as a general term covering the wide range of Open and proprietary operating systems commonly understood to be Unix ‘flavors’.

Easysoft and Easysoft Data Access are trademarks of Easysoft Limited.

CHAPTER 1 INTRODUCTION

Introducing the Easysoft ODBC-SQL Server Driver

The Easysoft ODBC-SQL Server Driver is an ODBC 3.51 driver for Microsoft SQL Server. It lets ODBC-enabled applications access SQL Server databases from Linux and Unix platforms. The Easysoft ODBC-SQL Server Driver supports:

- SQL Server 7.0
- SQL Server 2000
- SQL Server 2005
- SQL Server 2005 Express Edition

Chapter Guide

- **Product Status**
- **Deployment**

Product Status

The Easysoft ODBC-SQL Server Driver software is currently available on Unix and Linux platforms. The most up to date list of Easysoft ODBC-SQL Server Driver platforms is available at:

http://www.easysoft.com/products/data_access/odbc-sql-server-driver/index.html

Software problems can be reported to **support@easysoft.com** by users who have either purchased support or registered at the Easysoft web site at **http://www.easysoft.com** and are evaluating Easysoft products.

Deployment

The Easysoft ODBC-SQL Server Driver uses the Tabular Data Stream (TDS) data transfer protocol to communicate with SQL Server.

The TCP/IP protocol must be enabled in the SQL Server instance that you want to connect to.

CHAPTER 2 INSTALLATION

Installing the Easysoft ODBC-SQL Server Driver

This chapter explains how to install, license and remove the Easysoft ODBC-SQL Server Driver.

The installation instructions assume you are, or are able to consult with, a system administrator.

Chapter Guide

- **Obtaining the Easysoft ODBC-SQL Server Driver**
- **What to Install**
- **Installing the Easysoft ODBC-SQL Server Driver**
- **Uninstalling the Easysoft ODBC-SQL Server Driver**

Obtaining the Easysoft ODBC-SQL Server Driver

There are three ways to obtain the Easysoft ODBC-SQL Server Driver:

- The Easysoft web site is available 24 hours a day at **http://www.easysoft.com** and lets you download product releases and documentation.

  Choose **Download** from the Easysoft ODBC-SQL Server Driver section of the web site and then choose the platform release that you require.

  If you have not already done so, you will need to register at the web site to download Easysoft software.

- The Easysoft FTP site is available 24 hours a day at **ftp://ftp.easysoft.com** and lets you download free patches, upgrades, documentation and beta releases of Easysoft products, as well as definitive releases.

  Change to the `pub/sql_server` subdirectory and then choose the platform release that you require.

- You can order Easysoft software on CD. To do this, **contact us** by email, telephone or post.

What to Install

The name of the Easysoft ODBC-SQL Server Driver distribution file varies from platform to platform. The file name format is:

- `odbc-sqlserver-`*`x`*`.`*`y`*`.`*`z`*`-`*`platform`*`.tar.gz`

where *`x`* is the major version number, *`y`* is the minor version number and *`z`* is the build index, which is incremented when minor changes are made.

`platform` depends on the operating system distribution you require. File names may have this format:

- `odbc-sqlserver-`*`x`*`.`*`y`*`.`*`z`*`-`*`platform-variation`*`.tar`

where *`platform-variation`* refers to alternative versions available for a single platform.

Note	Select the highest release available for your platform within your licensed major version number (installing software with a different major version number requires a new Easysoft license).

Unix file names may also be suffixed with `.gz` for a gzipped archive, `.bz2` for a bzip2ed archive, or `.Z` for a compressed archive.

Note	If you download a Unix file with a Windows browser, the browser may strip the file name extension. For example, if you download a `.gz` file and the browser strips the file name extension, it may not be obvious that the file is gzipped. Use `file` *filename* to find out the file type of the downloaded file.

Caution!

As long as you stop all Easysoft software first (or software that uses the Easysoft drivers), it is safe to reinstall or upgrade the Easysoft ODBC-SQL Server Driver without uninstalling.

If you do uninstall, you should first back up any configuration data that you still need, as uninstalling some Easysoft products will result in this information being deleted (license details remain in place).

Installing the Easysoft ODBC-SQL Server Driver

These instructions show how to install the Easysoft ODBC-SQL Server Driver on Unix platforms. Please read this section carefully **before** installing the Easysoft ODBC-SQL Server Driver.

Before you Install

Requirements

To install the Easysoft ODBC-SQL Server Driver on Unix you need:

- The Bourne shell in `/bin/sh`. If your Bourne shell is not located there, you may need to edit the first line of the installation script.
- Various commonly used Unix commands such as:

  `grep`, `awk`, `test`, `cut`, `ps`, `sed`, `cat`, `wc`, `uname`, `tr`, `find`, `echo`, `sum`, `head`, `tee`, `id`

  If you do not have any of these commands, they can usually be obtained from the **Free Software Foundation**. As the `tee` command does not work correctly on some systems, the distribution includes a `tee` replacement.

- Depending on the platform, you will need up to 10 MB of temporary space for the installation files and up to 10 MB of free disk space for the installed programs. If you also install the unixODBC Driver Manager, these numbers increase by approximately 1.5 MB.
- For Easysoft Licensing to work, you must do one of the following:
  - Install the Easysoft ODBC-SQL Server Driver in `/usr/local/easysoft`.
  - Install the Easysoft ODBC-SQL Server Driver elsewhere and symbolically link `/usr/local/easysoft` to wherever you chose to install the software.

    The installation will do this automatically for you so long as you run the installation as someone with permission to create `/usr/local/easysoft`.
  - Install the Easysoft ODBC-SQL Server Driver elsewhere and set the `EASYSOFT_ROOT` environment variable.

    For more information about setting the `EASYSOFT_ROOT` environment variable, see **"Post installation" on page 43**.
- An ODBC Driver Manager. Easysoft ODBC-SQL Server Driver distributions include the unixODBC Driver Manager.

- You do not have to be the `root` user to install, but you will need permission to create a directory in the chosen installation path. Also, if you are not the `root` user, it may not be possible for the installation to:

  1. Register the Easysoft ODBC-SQL Server Driver with unixODBC.
  2. Create the example data source in the SYSTEM `odbc.ini` file.
  3. Update the dynamic linker entries (some platforms only).

  If you are not `root`, these tasks will have to be done manually later.

  Easysoft recommend you install all components as the `root` user.

What you can Install

This distribution contains:

- The Easysoft ODBC-SQL Server Driver.
- The Easysoft ODBC-SQL Server Driver with SSL Support.
- The unixODBC Driver Manager.

You will need an ODBC Driver Manager to use the Easysoft ODBC-SQL Server Driver from your applications. The distribution therefore contains the **unixODBC Driver Manager**. Most (if not all) Unix applications and interfaces support the unixODBC Driver Manager. For example, Perl DBD::ODBC, PHP, Python and so on.

You do not have to install the unixODBC Driver Manager included with this distribution. You can use an existing copy of unixODBC. For example, a version of unixODBC installed by another Easysoft product, a version obtained from your operating system vendor or one that you built yourself. However, as Easysoft ensure that the unixODBC distributed with the Easysoft ODBC-SQL Server Driver has been tested with that driver, we recommend you use it.

If you choose to use an existing unixODBC Driver Manager, the installation script will attempt to locate it. The installation script looks for the Driver Manager in the standard places. If you have installed it in a non-standard location, the installation script will prompt you for the location. The installation primarily needs unixODBC's `odbcinst` command to install drivers and data sources.

Where to Install

This installation needs a location for the installed files. The default location is `/usr/local`.

At the start of the installation, you will be prompted for an installation path. All files are installed in a subdirectory of your specified path called `easysoft`. For example, if you accept the default location `/usr/local`, the product will be installed in `/usr/local/easysoft` and below.

If you choose a different installation path, the installation script will try to symbolically link `/usr/local/easysoft` to the `easysoft` subdirectory in your chosen location. This allows us to distribute binaries with built in dynamic linker run paths. If you are not `root` or the path `/usr/local/easysoft` already exists and is not a symbolic link, the installation will be unable to create the symbolic link. For information about how to correct this manually, see **"Post Installation Steps for non-root Installations" on page 43**.

Note that you cannot license Easysoft products until either of the following is true:

- `/usr/local/easysoft` exists either as a symbolic link to your chosen installation path or as the installation path itself.
- You have set EASYSOFT_ROOT to *`installation_path`*`/easysoft`.

Changes Made to Your System

This installation script installs files in subdirectories of the path requested at the start of the installation, Depending on what is installed, a few changes may be made to your system:

1. If you choose to install the Easysoft ODBC-SQL Server Driver into unixODBC, unixODBC's `odbcinst` command will be run to add an entry to your `odbcinst.ini` file. You can locate this file with `odbcinst -j`. (`odbcinst` is in *`installation_path`*`/easysoft/unixODBC/bin`, if you are using the unixODBC included with this distribution.)

   The Easysoft ODBC-SQL Server Driver distribution includes two drivers, one with SSL Support (`Easysoft ODBC-SQL Server SSL`) and one without SSL support (`Easysoft ODBC-SQL Server`). To access SQL Server over an encrypted connection, you must use the SSL-compatible version of the driver. For more information about the Easysoft ODBC-SQL Server Driver with SSL Support, see **"Encrypting Connections to SQL Server" on page 94**.

   The `odbcinst.ini` entry for the Easysoft ODBC-SQL Server Driver will look similar to this:

```
[Easysoft ODBC-SQL Server]
Driver          = /usr/local/easysoft/sqlserver/lib/libessqlsrv.so
Setup          = /usr/local/easysoft/sqlserver/lib/libessqlsrvS.so
Threading       = 0
FileUsage       = 1
DontDLClose     = 1
UsageCount      = 1
```

The `odbcinst.ini` entry for the Easysoft ODBC-SQL Server Driver with SSL Support will look similar to this:

```
[Easysoft ODBC-SQL Server SSL]
Driver      = /usr/local/easysoft/sqlserver/lib/libessqlsrv_ssl.so
Setup       = /usr/local/easysoft/sqlserver/lib/libessqlsrvS.so
Threading   = 0
FileUsage   = 1
DontDLClose = 1
UsageCount  = 1
```

For information about removing these entries, see **"Uninstalling the Easysoft ODBC-SQL Server Driver" on page 48**.

2. The installation script installs example data sources into unixODBC. The data sources will be added to your SYSTEM `odbc.ini` file. You can locate your SYSTEM `odbc.ini` file by using `odbcinst -j`. The data source for the standard driver will look similar to this:

```
[SQLSERVER_SAMPLE]
Driver          = Easysoft ODBC-SQL Server
Description     = Easysoft SQL Server ODBC driver
Server          = server.domain.com
Port            = 1422
Database        = northwind
User            = sa
Password        = password
Mars_Connection = No
Logging         = 0
LogFile         =
QuotedId        = Yes
AnsiNPW         = Yes
Language        =
Version7        = No
```

The data source for the Easysoft ODBC-SQL Server Driver with SSL Support will look similar to this:

```
[SQLSERVER_SAMPLE_SSL]
Driver                  = Easysoft ODBC-SQL Server SSL
Description             = Easysoft SQL Server ODBC driver
Server                  = server.domain.com
Port                    = 1433
Database                = northwind
User                    = sa
Password                = password
Mars_Connection         = No
Logging                 = 0
LogFile                 =
QuotedId                = Yes
AnsiNPW                 = Yes
Language                =
Version7                = No
Encrypt                 = Yes
TrustServerCertificate  = No
PrivateKeyFile          =
CertificateFile         =
Entropy                 =
```

For information about removing these data sources, see **"Uninstalling the Easysoft ODBC-SQL Server Driver" on page 48**.

3. Dynamic Linker.

   On operating systems where the dynamic linker has a file listing locations for shared objects (Linux), the installation script will attempt to add paths under the path you provided at the start of the installation to the end of this list. On Linux, this is usually the file `/etc/ld.so.conf`.

Reinstalling or Installing When You Already Have Other Easysoft Products Installed

Each Easysoft distribution contains common files shared between Easysoft products. These shared objects are placed in *`installation_path`*`/easysoft/lib`. When you run the installation script, the dates and versions of these files will be compared with the same files in the distribution. The files are only updated if the files being installed are newer or have a later version number.

You should ensure that nothing on your system is using Easysoft software before starting an installation. This is because on some platforms, files in use cannot be replaced. If a file cannot be updated, you will see a warning during the installation. All warnings are written to a file called `warnings` in the directory you unpacked the distribution into.

If the installer detects you are upgrading a product, the installer will suggest you delete the product directory to avoid having problems with files in use. An alternative is to rename the specified directory.

If you are upgrading, you will need a new license from Easysoft to use the new driver.

Gathering Information Required During the Installation

During the installation, you will be prompted for various pieces of information. Before installing, you need to find out whether you have unixODBC already installed and where it is installed. The installation script searches standard places like `/usr` and `/usr/local`. However, if you installed the Driver Manager in a non-standard place and you do not install the included unixODBC, you will need to know the location.

If you want to use the installation to test the connection to your SQL Server machine and interactively create an ODBC data source, you will need:

- The host name of the SQL Server machine.
- The name of an instance to connect to or the port that the instance is listening on.
- A valid SQL Server login name and password that can be used to connect to the database you want to access.

INSTALLATION

Unpacking the Distribution

The distribution for Unix platforms is a tar file. There are multiple copies of the same distribution with different levels of compression. You unpack the distribution in one of the following ways.

If the distribution file has been gzipped (`.gz`), use:

```
gunzip odbc-sqlserver-x.y.z-platform.tar.gz
```

If the distribution file has been bzipped (`.bz2`), use:

```
bunzip2 odbc-sqlserver-x.y.z-platform.tar.bz2
```

If the distribution file has been compressed, (`.Z`), use:

```
uncompress odbcsql-server-x.y.z-platform.tar.Z
```

You may have a distribution file which is not compressed at all (`.tar`). To extract the installation files from the tar file, use:

```
tar -xvf odbc-sqlserver-x.y.z-platform.tar
```

This will create a directory with the same name as the tar file (without the `.tar` postfix) containing further archives, checksum files, an installation script and various other installation files.

Change directory into the directory created by unpacking the tar file.

License to Use

The End-User License Agreement is contained in the file `license.txt`. Be sure to understand the terms of the agreement before continuing, as you will be required to accept the license terms at the start of the installation.

Answering Questions During the Installation

Throughout the installation, you will be asked to answer some questions. In each case, the default choice will be displayed in square brackets and you need only press Enter to accept the default. If there are alternative responses, these will be shown in round brackets; to choose one of these, type the response and press Enter.

For example:

```
Do you want to continue? (y/n) [n]:
```

The possible answers to this question are y or n. The default answer when you type nothing and press Enter is n.

Running the Installer

Before you run the installer, make sure you have read **"Installation" on page 28**. If you are considering running the installation as a non `root` user, we suggest you review this carefully as you will have to get a `root` user to manually complete some parts of the installation afterwards. Easysoft recommend installing as the `root` user. (If you are concerned about the changes that will be made to your system, see **"Changes Made to Your System" on page 23**.)

To start the installation, run:

```
./install
```

You will need to:

- Confirm your acceptance of the license agreement by typing "yes" or "no".

  For more information about the license agreement, see **"License to Use" on page 29**.

- Supply the location where the software is to be installed. Easysoft recommend accepting the default installation path.

  For more information, see **"Where to Install" on page 22**.

> **Note** If you are upgrading, you will need a new license from Easysoft.

Locating or Installing unixODBC

Easysoft strongly recommend you use the unixODBC Driver Manager because:

- The installation script is designed to work with unixODBC and can automatically add Easysoft ODBC-SQL Server Driver and data sources during the installation.
- Most ODBC-enabled applications and interfaces support unixODBC. The Easysoft ODBC-SQL Server Driver and any data sources that you add during the installation will automatically be available to your applications and interfaces therefore.
- The unixODBC project is currently led by Easysoft developer Nick Gorham. This means that there is a great deal of experience at Easysoft of unixODBC in general and of supporting the Easysoft ODBC-SQL Server Driver running under unixODBC. It also means that if you find a problem in unixODBC, it is much easier for us to facilitate a fix.

The installation starts by searching for unixODBC. There are two possible outcomes here:

1. If the installation script finds unixODBC, the following message will be output:

   ```
   Found unixODBC under /unixODBC_path
      and it is version n.n.n
   ```

2. If the installation script cannot find unixODBC in the standard places, you will be asked whether you have it installed.

If unixODBC is installed, you need to provide the unixODBC installation path. Usually, the path required is the directory above where `odbcinst` is installed. For example, if `odbcinst` is in `/opt/unixODBC/bin/odbcinst`, the required path is `/opt/unixODBC`.

If unixODBC is not installed, you should install the unixODBC included with this distribution.

If you already have unixODBC installed, you do not have to install the unixODBC included with the distribution, but you might consider doing so if your version is older than the one included.

The unixODBC in the Easysoft ODBC-SQL Server Driver distribution is not built with the default options in unixODBC's configure line.

Option	Description
--prefix=/etc	This means the default SYSTEM `odbc.ini` file where SYSTEM data sources are located will be `/etc/odbc.ini`.
--enable-drivers=no	This means other ODBC drivers that come with unixODBC are not installed.
--enable-iconv=no	This means unixODBC will not look for a libiconv. Warnings about not finding an iconv library were confusing our customers.

Option	Description
--enable-stats=no	Disables unixODBC statistics, which use system semaphores to keep track of used handles. Many systems do not have sufficient semaphore resources to keep track of used handles. In addition, the statistics are only available in the GUI ODBC Administrator.
--enable-readline=no	This disables readline support in `isql`. We disabled this because it ties `isql` to the version of libreadline on the system we build on. We build on as old a version of the operating system as we can for forward compatibility. Many newer Linux systems no longer include the older readline libraries and so enabling readline support makes `isql` unusable on these systems.
--prefix=/usr/local/easysoft/unixODBC	This installs unixODBC into `/usr/local/easysoft/unixODBC`.

Figure 1: Easysoft unixODBC configure line options.

Installing the Easysoft ODBC-SQL Server Driver

The Easysoft ODBC-SQL Server Driver installation script:

- Installs the driver.
- Registers the driver with the unixODBC Driver Manager.

  If the Easysoft ODBC-SQL Server Driver is already registered with unixODBC, a warning will be displayed that lists the drivers unixODBC knows about. If you are installing the Easysoft ODBC-SQL Server Driver into a different directory than it was installed before, you will need to edit your `odbcinst.ini` file after the installation and correct the Driver and Setup paths. unixODBC's `odbcinst` will not update these paths if a driver is already registered.
- Creates an example Easysoft ODBC-SQL Server Driver data source.

  If unixODBC is installed and you registered the Easysoft ODBC-SQL Server Driver with unixODBC, an example data source will be added to your `odbc.ini` file.

  If a data source called "SQLSERVER_SAMPLE" already exists, the existing data source will be displayed and you have the option to replace it.

Licensing

The *`installation_path`*`/easysoft/license/licshell` program lets you obtain or list licenses.

Licenses are stored in the *`installation_path`*`/easysoft/license/licenses` file. After obtaining a license, you should make a backup copy of this file.

The installation script asks you if you want to request an Easysoft ODBC-SQL Server Driver license:

```
Would you like to request a Easysoft ODBC-SQL
Server Driver license now (y/n) [y]:
```

You do not need to obtain a license during the installation, you can run `licshell` after the installation to obtain or view licenses.

If you answer yes, the installation runs the `licshell` script. The process of obtaining a license is best described in the **Licensing Guide**.

To obtain a license automatically, you will need to be connected to the Internet and allow outgoing connections to `license.easysoft.com` on port 8884. If you are not connected to the Internet or do not allow outgoing connections on port 8884, the License Client can create a license request file that you can mail or fax to Easysoft. You can also supply the details to us by telephone.

Start the License Client. The following menu is displayed:

```
[0] exit

[1] view existing license

[n] obtain a license for the desired product.
```

To obtain a license, select one of the options from `[2]` onwards for the product you are installing. The License Client will then run a program that generates a key that is used to identify the product and operating system (we need this key to license you).

After you have chosen the product to license (Easysoft ODBC-SQL Server Driver), you need to supply:

- Your full name.
- Your company name.
- An email contact address. This **must** be the email address that you used when you registered on the Easysoft web site.
- Your telephone number (you need to specify this if you telephone us to request a license).
- Your fax number (you need to specify this if you fax the license request to us).
- A reference number. When applying for a trial license, just press Enter when prompted for a reference number. This field is used to enter a reference number that we will supply you for full (paid) licenses.

You will then be asked to specify how you want to obtain the license. The choices are:

```
[1] Automatically by contacting the Easysoft
License Daemon
```

This requires a connection to the Internet and the ability to support an outgoing TCP/IP connection to `license.easysoft.com` on port 8884.

```
[2] Write information to file so you can fax,
telephone it
```

The license request is output to `license_request.txt`.

```
[3] Cancel this operation
```

If you choose to obtain the license automatically, the License Client will start a TCP/IP connection to `license.easysoft.com` on port 8884 and send the details you supplied and your site number. No other data is sent. The data sent is transmitted as plain text, so if you want to avoid the possibility of this information being intercepted by someone else on the Internet, you should choose `[2]` and telephone or fax the request to us. The License daemon will return the license key, print it to the screen and make it available to the installation script in the file `licenses.out`.

If you choose option `[2]`, the license request is written to the file `license_request.txt`. You should then exit the License Client by choosing option `[0]` and complete the installation. After you have mailed, faxed or telephoned the license request to us, we will return a license key. Add this to the end of the file *`installation_path`*`/easysoft/license/licenses`.

If any warnings or errors are output during this process, please mail the output to **support@easysoft.com** and we will correct the problem.

Testing the Connection to SQL Server

The Easysoft ODBC-SQL Server Driver installation lets you test the connection to SQL Server, save the connection settings in an ODBC data source and retrieve some SQL Server data. Although the installation default is to do this test, you do not have to.

The installation guides you through the connection process step by step, using `tdshelper` (a diagnostic program supplied with the Easysoft ODBC-SQL Server Driver) to test the SQL Server connection and check that you can access SQL Server with your login name and password. If at any time you want to stop the test, type q at any prompt.

If you decide to skip this part of the installation, you can use `tdshelper` after the installation completes to check your SQL Server connection settings. The installation script installs `tdshelper` in the *`installation_path`*`/easysoft/sqlserver/bin` directory. For more information about running `tdshelper`, see **"Testing the Connection to SQL Server with tdshelper" on page 52**.

The installation uses `tdshelper` to search for SQL Server instances that are listening on your network. The results of a successful search will look similar to this:

```
Using /usr/local/easysoft/sqlserver/bin/tdshelper -i -c 1
============================================================
    ServerName MYSQLSERVER2000HOST              Port 1433 (Default)
    ServerName MYSQLEXPRESSHOST\SQLEXPRESS        Port 2777
    ServerName MYSQLSERVER2005HOST\MYINSTANCEI    Port 1510
    ServerName MYSQLSERVER2005HOST\MYINSTANCEII   Port 1511
============================================================
```

If you do not see the SQL Server instance that you want to connect to in the list or the list is empty, the SQL Server Browser or SQL Server 2000 listener service may not be running. `tdshelper` uses the SQL Server Browser or listener to find out the available SQL Server instances. If the browser or listener is not running, the installation will be unable to use `tdshelper` to help you interactively connect to SQL Server and create a data source. Type q to exit and then manually create a data source after the installation completes. The installation creates a sample data source that you can use as a starting point when setting up your own Easysoft ODBC-SQL Server Driver data sources. For more information about creating data sources, see **"Setting Up Data Sources" on page 61**.

The example output shows that:

- The default SQL Server instance on a machine named MYSQLSERVER2000HOST is listening on the default SQL Server TCP port 1433.
- The default named SQL Server Express instance on a machine named MYSQLEXPRESSHOST is listening on port 2777.
- There are two named instances running on MYSQLSERVER2005HOST. The instances are listening on ports 1510 and 1511 respectively.

If the SQL Server instance that you want to connect to is listed in the results, type y to continue interactively creating your SQL Server data source.

If you chose to continue, type the name (or IP address) of the machine where your SQL Server instance is running when prompted. To connect to a named instance, use the format *machinename**instancename*. To connect to a SQL Server Express instance, use the format *machinename*\SQLEXPRESS. To connect to a SQL Server instance that is not listening on the default port (1433), use the format *machinename:port.*

Based on the example output shown earlier, you would type:

- `MYSQLSERVERHOST` to connect to the default instance on this machine.
- `MYSQLEXPRESSHOST\SQLEXPRESS` to connect to the SQL Server Express instance.
- `MYSQLSERVER2005HOST:1510` to connect to the first named instance on this machine and `MYSQLSERVER2005HOST:1511` to connect to the second.

Type your SQL Server login name when prompted. If you usually connect to SQL Server through your Windows account, type your Windows user name. If the SQL Server instance is running on a machine that is part of a Windows domain, use the format *domain**username*.

Otherwise, type a valid SQL Server user name.

Type the password for your user name when prompted.

If `tdshelper` can successfully connect to the SQL Server instance, a list of databases that you can access is displayed.

When setting up your SQL Server login, your database administrator will have associated a database with your login. This is the default database for the connection. The default database is listed first in the `tdshelper` output. If you want to connect to a different database, type the name of another databases in the list. Otherwise, press RETURN to connect to the default database.

If you want to change the language of SQL Server system messages, type one of the listed languages when prompted. Otherwise, press RETURN to accept the default language (again, this is listed first in the `tdshelper` output).

The Easysoft ODBC-SQL Server Driver installation has now gathered enough information to connect to SQL Server. The installation lets you save this connection information in an ODBC data source. You can use this data source to connect to SQL Server now and when the installation completes. The data source is written to your system `odbc.ini` file.

Finally, the installation prompts you whether to retrieve a list of tables from the SQL Server database. The installation uses unixODBC's `isql` and your new data source to do this. Note that if you chose not to license the Easysoft ODBC-SQL Server Driver earlier in the installation, skip this step. The Easysoft ODBC-SQL Server Driver needs to be licensed before it can be used to connect to a data source. When the installation has finished, you can use `isql` to test the data source after you have licensed the Easysoft ODBC-SQL Server Driver.

POST INSTALLATION

Supplied Documents and Examples

The last part of the installation runs a post install script that lists the resources available to you.

- The Easysoft ODBC-SQL Server Driver documentation is installed in *installation_path*/easysoft/sqlserver/doc:
  - The Easysoft ODBC-SQL Server Driver manual in PDF format.
  - The Easysoft ODBC-SQL Server Driver EULA.

installation_path/easysoft/sqlserver/doc/CHANGES.txt lists all the changes in each version of the Easysoft ODBC-SQL Server Driver.

There are also many resources at the **Easysoft web site**.

Post Installation Steps for non-root Installations

If you installed the Easysoft ODBC-SQL Server Driver as a non-root user (not recommended), there may be some additional steps you will need to do manually:

1. If you attempt to install the Easysoft ODBC-SQL Server Driver under the unixODBC Driver Manager and you do not have write permission to unixODBC's odbcinst.ini file, the driver cannot be added.

   You can manually install the driver under unixODBC by adding an entry to the odbcinst.ini file. Run odbcinst -j to find out the location of the DRIVERS file then append the lines from the drv_template file to the odbcinst.ini file. (drv_template is in the directory where the distribution was untarred to)

2. No example data sources can be added into unixODBC if you do not have write permission to the SYSTEM `odbc.ini` file. Run `odbcinst -j` to find out the location of the `SYSTEM DATA SOURCES` file then add your data sources to this file.

3. On systems where the dynamic linker has a configuration file defining the locations where it looks for shared objects (Linux), you will need to add:

   ```
   installation_path/easysoft/lib
   installation_path/easysoft/unixODBC/lib
   ```

   The latter entry is only required if you installed the unixODBC included with this distribution. Sometimes, after changing the dynamic linker configuration file, you need to run a program to update the dynamic linker cache. (For example, `/sbin/ldconfig` on Linux.)

4. If you did not install the Easysoft ODBC-SQL Server Driver in the default location, you need to do one of the following:

   - Link `/usr/local/easysoft` to the `easysoft` directory in your chosen installation path.

     For example, if you installed in `/home/user`, the installation will create `/home/user/easysoft` and you need to symbolically link `/usr/local/easysoft` to `/home/user/easysoft`:

     ```
     ln -s /home/user/easysoft /usr/local/easysoft
     ```

   - Set and export the `EASYSOFT_ROOT` environment variable to *`installation_path`*`/easysoft`.

5. If your system does not have a dynamic linker configuration file, you need to add the paths listed in step 3 to whatever environment path the dynamic linker uses to locate shared objects. You may want to amend this in a system file run whenever someone logs in such as `/etc/profile`.

   The environment variable depends on the dynamic linker. Refer to your `ld` or `ld.so` man page. It is usually:

   `LD_LIBRARY_PATH`, `LIBPATH`, `LD_RUN_PATH` or `SHLIB_PATH`.

Setting Dynamic Linker Search Paths

Your applications will be linked against an ODBC Driver Manager, which will load the ODBC Driver you require. The dynamic linker needs to know where to find the ODBC Driver Manager shared object. The ODBC Driver Manager will load the Easysoft ODBC-SQL Server Driver, which is dependent on further common Easysoft shared objects; the dynamic linker needs to locate these too.

On operating systems where the dynamic linker has a file specifying locations for shared objects (Linux), the installation will attempt to add paths under the path you provided at the start of the installation to the end of this list; no further action should be required. For more information, see **"Dynamic Linker." on page 27**.

On other Unix platforms, there are two methods of telling the dynamic linker where to look for shared objects:

1. You add the search paths to an environment variable and export it.

   This method always works and overrides the second method, described below.

2. At build time, a run path is inserted into the executable or shared objects. On System V systems, Easysoft mostly distribute Easysoft ODBC-SQL Server Driver shared objects with an embedded run path. The dynamic linker uses the run path to locate Easysoft ODBC-SQL Server Driver shared object dependencies.

   For the first method, the environment variable you need to set depends on the platform (refer to the platform documentation for `ld(1)`, `dlopen` or `ld.so(8)`).

Environment Variable	Platform
LD_LIBRARY_PATH	System V based operating systems and Solaris.
LIBPATH	AIX
SHLIB_PATH	HP-UX
LD_RUN_PATH	Many platforms use this in addition to those listed above.

Figure 2: Dynamic linker search path environment variables.

To use the Easysoft ODBC-SQL Server Driver, you need to add:

`installationdir``/easysoft/sqlserver:`*`installationdir`*`/easysoft/lib`

where *`installationdir`* is the directory in which you chose to install the Easysoft ODBC-SQL Server Driver. If you accepted the default location, this is `/usr/local`.

An example of setting the environment path in the Bourne shell on Solaris is:

```
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/easysoft/sqlserver:/usr/local/easysoft/lib

export LD_LIBRARY_PATH
```

> **Note** The exact command you need to set and export an environment variable depends on your shell.

If you installed the unixODBC Driver Manager included in the Easysoft ODBC-SQL Server Driver distribution, you also need to add *`installationdir`*`/easysoft/unixODBC/lib` to the dynamic linker search path.

Uninstalling the Easysoft ODBC-SQL Server Driver

There is no automated way to remove the Easysoft ODBC-SQL Server Driver in this release. However, removal is quite simple. To do this, follow these instructions.

To uninstall the Easysoft ODBC-SQL Server Driver

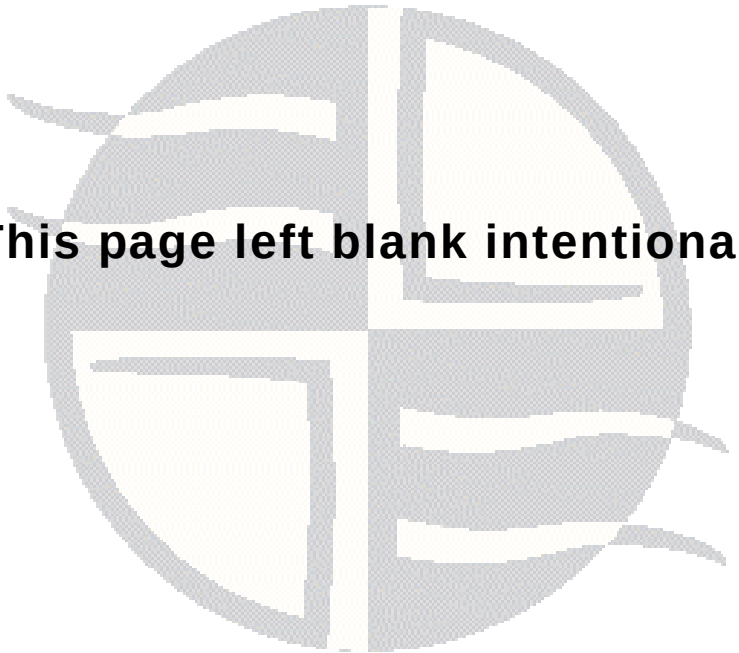
1. Change directory to *`installation_path`*`/easysoft` and delete the `sqlserver` directory. *`installation_path`* is the Easysoft ODBC-SQL Server Driver installation directory, by default `/usr/local`.
2. If you had to add this path to the dynamic linker search paths (for example, `/etc/ld.so.conf` on Linux), remove it. You may have to run a linker command such as `/sbin/ldconfig` to get the dynamic linker to reread its configuration file. Usually, this step can only be done by the `root` user.
3. If you were using unixODBC, the Easysoft ODBC-SQL Server Driver entry needs to be removed from the `odbcinst.ini` file. To check whether the Easysoft ODBC-SQL Server Driver is configured under unixODBC, use `odbcinst -q -d`. If the command output contains `[Easysoft ODBC-SQL Server Driver]` and `[Easysoft ODBC-SQL Server Driver SSL]`, uninstall the drivers from unixODBC by using:

   ```
   odbcinst -u -d -n 'Easysoft ODBC-SQL Server'
   ```

   ```
   odbcinst -u -d -n 'Easysoft ODBC-SQL Server SSL'
   ```

   If a reduced usage count message is displayed, repeat this command until `odbcinst` reports that the drivers have been removed.

4. If you created any Easysoft ODBC-SQL Server Driver data sources under unixODBC, you may want to delete these. To do this, first use `odbcinst -j` to locate USER and SYSTEM `odbc.ini` files. Then check those files for data sources that have the driver attribute set to `Easysoft ODBC-SQL Server`.
5. Remove `sqlserver_install.info` from the `/usr/local/easysoft` directory.



This page left blank intentionally

CHAPTER 3 CONFIGURATION

Configuring the Easysoft ODBC-SQL Server Driver

The Easysoft ODBC-SQL Server Driver is installed on the computer where your applications are running. ODBC applications access ODBC drivers through the ODBC Driver Manager and a data source. The data source tells the Driver Manager which ODBC driver to load, which SQL Server instance to connect to and how to connect to it. This chapter describes how to create data sources, use DSN-less connections and configure the Easysoft ODBC-SQL Server Driver.

Before setting up a data source on your computer, you must have successfully installed the Easysoft ODBC-SQL Server Driver on this computer.

For Easysoft ODBC-SQL Server Driver installation instructions, see **"Installation" on page 15**.

Chapter Guide

- **Testing the Connection to SQL Server with tdshelper**
- **Configuring the Easysoft ODBC-SQL Server Driver**
- **Setting Up Data Sources**
- **DSN-less Connections**

Testing the Connection to SQL Server with tdshelper

The Easysoft ODBC-SQL Server Driver distribution includes the diagnostic program `tdshelper`. `tdshelper` lets you:

- Search for SQL Server instances on your network.
- Test the connection to an instance.
- Check that you access an instance with your SQL Server login.
- List the databases that are available to your login.

The Easysoft ODBC-SQL Server Driver installation script installs `tdshelper` in `/usr/local/easysoft/sqlserver/bin`.

TDSHELPER COMMAND LINE OPTIONS

`tdshelper` has the following command line:

`tdshelper [`*`options`*`]`

The options are:

`-a --auth` *`password`*

The password for the SQL Server login name specified with the `-u` option.

`-c --count` *`number`*

The number of probe packets to send when searching for SQL Server instances. The default is 5.

`-d --dblist`

List the databases that are available to the login name specified by `-u` in the SQL Server instance specified by `-s`. The default database for the login is shown first in the command output.

`-l --langlist`

List the languages that SQL Server system messages are available in. To retrieve the languages, you also need to specify a SQL Server instance (`-s`) and login (`-u -a`). The default language for the login is shown first in the command output.

`-n --port` *`port`*

The TCP port that the SQL Server instance specified with the `-s` option is listening on.

If you omit the port and are connecting to a named instance, `tdshelper` will use the SQL Server Browser or listener to detect the port. If you omit the port and are connecting to the default instance, `tdshelper` will try to connect to port 1433.

`-p --probe`

Search for SQL Server instances on the network. For each instance found, the command's output displays:

- The IP address of the machine the instance is running on.
- The port the instance is listening on.
- The host name of the machine the instance is running on.
- The instance name. The name used to identify the default SQL Server instance is `MSSQLSERVER`.
- The SQL Server version number.

`-i --instance`

Search for SQL Server instances on the network. The output is more concise than that produced by the `-p` option. The output lists each instance found using the format *`machinename`*`\`*`instancename`*. Default instances are listed as *`machinename`*.

> **Note** For `tdshelper` to be able to list available instances (`-p` or `-i`), the SQL Server Browser or the SQL Server 2000 listener service must be running.

`-s --server` *`instance`*

The SQL Server instance to connect to. To test the connection to the default SQL Server instance, replace *`instance`* with the host name or IP address of the machine where the instance is running. To test the connection to a named instance, replace *instance* with *machinename**instance*, where:

- *machinename* is the host name or IP address of the SQL Server machine
- *instancename* is the name of the instance that you want to connect to.

If the SQL Server Browser or listener service is not running and the instance is not listening on the default port 1433, specify the port with `-p`.

To test database authentication, you also need to specify a valid SQL Server login name and password for the instance with `-u` and `-a`.

`-u --user` *`login`*

A valid SQL Server login name for the instance specified with `-s`. If you usually connect to SQL Server through your Windows account, type your Windows user name. If the SQL Server instance is running on a machine that is part of a Windows domain, use the format *domain**username*. Otherwise, specify a valid SQL Server user name. Use `-a` to specify the password.

> **Note**
>
> You may have to enclose the SQL Server instance name, login name and password with single quotes. Do this to protect any backslashes, spaces or other special characters these settings contain. For example:
>
> ```
> ./tdshelper -s 'mymachine\myinstance' -v
> ```

`-v --verbose`

Display extra status and diagnostic information. Use this option with `-s` to diagnose connection problems.Testing your SQL Server Connection Settings

To connect to a SQL Server instance, you need this information:

- The hostname of the machine where the instance is running.
- The instance name if it is not the default instance.
- A SQL Server login name and password for the instance.

`tdshelper` lets you confirm that your connection settings are valid before you use the information in an Easysoft ODBC-SQL Server Driver data source. If the SQL Server Browser or listener is running, you can also use `tdshelper` to verify that the SQL Server instance is running and visible from the machine where the Easysoft ODBC-SQL Server Driver is installed.

1. `cd /usr/local/easysoft/sqlserver/bin`
2. If the SQL Server Browser or listener is running, check that your SQL Server instance is running by typing the following command. Otherwise, skip this step.

   ```
   ./tdshelper -i -c 1
   ```

   If the instance is running, it will be listed in the `tdshelper` output. In the following example output, the default SQL Server instance is running on the machine MYSQLSERVER2000HOST; the default named SQL Server Express instance is running on the machine MYSQLEXPRESSHOST; a named SQL Server instance is running on the machine MYSQLSERVER2005HOST:

```
ServerName MYSQLSERVER2000HOST                Port 1433 (Default)
ServerName MYSQLEXPRESSHOST\SQLEXPRESS        Port 2777
ServerName MYSQLSERVER2005HOST\MYINSTANCEI    Port 1510
```

   The output also shows the TCP port that each instance is listening on.

3. Do one of the following:

   - To test the connection to a default SQL Server instance, type:

   ```
   ./tdshelper -s machinename -n port -v
   ```

   Replace *machinename* with the host name of the SQL Server machine. Replace *port* with the TCP port on which the instance is listening. For example, `tdshelper -s MYSQLSERVER2000HOST -n 1433 -v`.

   - To test the connection to a named SQL Server instance, type:

   ```
   ./tdshelper -s 'machinename\instance' -n port -v
   ```

   Replace *machinename* with the host name of the SQL Server machine and *instance* with the instance name. Replace *port* with the TCP port on which the instance is listening. For example, `tdshelper -s 'MYSQLSERVER2005HOST\MYINSTANCEI' -n 1510 -v`.

   If `tdshelper` can connect to SQL Server, you will see output similar to this:

   ```
   tdshelper: connecting to MYSQLSERVER2000HOST on port 1433

   tdshelper: succesfully opened connection
   ```

   When configuring an Easysoft ODBC-SQL Server Driver data source to connect to this instance, use the value you specified with the `-s` option for the `Server` attribute value.

If `tdshelper` is unable to connect and displays a `"Connection refused"` error, the SQL Server instance may not be running, be listening on the port you specified or allow remote TCP/IP connections. For more information, see **"Client unable to establish connection: OS Error: 'Connection refused'" on page 76** and **"Client unable to establish connection: Server not configured for TCP connection" on page 77**.

4. To check that you can access the instance with your SQL Server login name and password, use the `-u` and `-a` options.

   If you usually connect to SQL Server through your Windows account, specify your Windows user name with `-u`. If the SQL Server instance is running on a machine that is part of a Windows domain, use the format *domain**username*. Otherwise, specify a valid SQL Server user name.

   Specify the password for your user name with `-a`.

   For example:

   ```
   ./tdshelper -s 'MYSQLSERVER2005HOST\MYINSTANCEI' -u
   'mydomain\mywindowsuser' -a mywindowspassword -v
   ```

   – OR –

   ```
   /tdshelper -s MYSQLSERVER2000HOST -u
   mysqlserveruser -a mysqlserverpassword -v
   ```

   If `tdshelper` can successfully access the SQL Server instance with your user name and password, the output will contain the following line:

   ```
   tdshelper: succesfully logged into server with
   diagnostic records
   ```

When configuring an Easysoft ODBC-SQL Server Driver data source to connect to this instance, the `User` and `Password` attribute values should be the same as the values you specified with the `-u` and `-a` options.

If `tdshelper` is unable to connect and displays a "`Login failed for user ''. The user is not associated with a trusted SQL Server connection.`" error, check that you specified a valid Windows user name and password. If `tdshelper` fails with an "`Login failed for user 'myuser'.`" error, check that you specified a valid SQL Server user name and password.

For information about other situations in which these errors display, see **"Login failed for user ". The user is not associated with a trusted connection" on page 79** and **"Login failed for user 'myuser'." on page 80**.

Configuring the Easysoft ODBC-SQL Server Driver

This section describes how to configure the Easysoft ODBC-SQL Server Driver to connect to a SQL Server database by using a data source or a DSN-less connection string. The section assumes you are, or are able to consult with, a system administrator.

Setting Up Data Sources

There are two ways to set up a data source to your SQL Server data:

- Create a SYSTEM data source, which is available to anyone who logs on to this Unix machine.

– OR –

- Create a USER data source, which is only available to the user who is currently logged on to this Unix machine.

By default, the Easysoft ODBC-SQL Server Driver installation creates a SYSTEM data source named `[SQLSERVER_SAMPLE]`. If you are using the unixODBC included in the Easysoft ODBC-SQL Server Driver distribution, the SYSTEM `odbc.ini` file is in `/etc`.

If you built unixODBC yourself, or installed it from some other source, SYSTEM data sources are stored in the path specified with the configure option `--sysconfdir=`*`directory`*. If `sysconfdir` was not specified when unixODBC was configured and built, it defaults to `/usr/local/etc`.

If you accepted the default choices when installing the Easysoft ODBC-SQL Server Driver, USER data sources must be created and edited in `$HOME/.odbc.ini`.

Note

To display the directory where unixODBC stores SYSTEM and USER data sources, type `odbcinst -j`.

By default, you must be logged in as `root` to edit a SYSTEM data source defined in `/etc/odbc.ini`.

You can either edit the sample data source or create new data sources.

Each section of the `odbc.ini` file starts with a data source name in square brackets [ ] followed by a number of *attribute=value* pairs.

> **Note** Attribute names in `odbc.ini` are not case sensitive.

The `Driver` attribute identifies the ODBC driver in the `odbcinst.ini` file to use for a data source. The Easysoft ODBC-SQL Server Driver distribution includes two drivers:

- One with SSL support that should be used if you need to access SQL Server 2000 or 2005 over an encrypted connection.
- One without SSL support that should be used for SQL Server 2000 and 2005 when encryption is not required and for SQL Server 7.0.

When the Easysoft ODBC-SQL Server Driver is installed into unixODBC, entries for the standard driver (`Easysoft ODBC-SQL Server`) and the driver with SSL support (`Easysoft ODBC-SQL Server SSL`) are placed in `odbcinst.ini`.

For Easysoft ODBC-SQL Server Driver data sources, you need to include a `Driver = Easysoft ODBC-SQL Server` entry.

For Easysoft ODBC-SQL Server Driver with SSL Support data sources, you need to include a `Driver = Easysoft ODBC-SQL Server SSL` entry. For more information about configuring Easysoft ODBC-SQL Server Driver with SSL Support data sources, see **"Encrypting Connections to SQL Server" on page 94**.

To configure a SQL Server data source, in your `odbc.ini` file, you need to specify:

- The host name or IP address of the machine where the SQL Server instance is running. To connect to a named instance you also need to specify the instance name. (`Server`)
- A valid SQL Server login name (`User`) and password (`Password`).

For example:

```
[SQL Server]
Driver    = Easysoft ODBC-SQL Server
# To connect to the default instance, omit \my_instance_name.
Server    = my_sqlserver_hostname\my_instance_name
User      = my_user
Password  = my_password
```

If the SQL Server Browser or listener service is not in use at your site and you want to connect to an instance that is not listening on the default TCP port (1433), you also need to specify the port: For example, to connect to a SQL Server instance that is listening on port 1500, add this entry:

```
Port              = 1500
```

ATTRIBUTE FIELDS

The following attributes may be set in the `odbc.ini` file:

Attribute	Description
Driver = *value*	The name of the ODBC driver to use with this data source. To connect to a SQL Server 2000 or 2005 instance over an encrypted connection, set this attribute value to `Easysoft ODBC-SQL Server SSL`. Otherwise, set this attribute value to `Easysoft ODBC-SQL Server`.
Description = *value*	A single line of descriptive text that may be retrieved by some applications to describe the data source.
Server = *value*	The SQL Server instance that you want to connect to. To connect to the default SQL Server instance, type: *machinename* where *machinename* is the name or IP address of the host where SQL Server is running. To connect to a named instance, type: *machinename**instancename* where *instancename* is the SQL Server instance. To connect to the default SQL Server Express named instance, type: *machinename*\sqlexpress If the SQL Server Browser or listener service is running, you can use `/usr/local/easysoft/sqlserver/bin/tdshelper` to display a list of available SQL Server instances: `./tdshelper -i -c 1`

Attribute	Description
Port = *num*	The TCP port that SQL Server is listening on. If you are connecting to a default instance that is listening on port 1433, the `Port` setting can be omitted. If the SQL Server Browser or the SQL Server 2000 listener service is running, the Easysoft ODBC-SQL Server Driver will automatically detect the port number and the `Port` setting can be omitted. By default, named instances of SQL Server use dynamic ports, which means that an available port is assigned when the instance starts. If a SQL Server instance is listening on a dynamically allocated port number, you must omit the `Port` setting and let the Easysoft ODBC-SQL Server Driver use the browser or listener to detect the port number. If the SQL Server Browser or listener is not running at your site, your database administrator will have configured each SQL Server instance to listen on a specific TCP port. You need to specify this port with the `Port` setting. If your database administrator has hidden the SQL Server instance from the SQL Server Browser or listener, you need to specify the port number of the hidden instance. If your database administrator has configured the SQL Server instance to listen on multiple ports, use the `Port` setting to specify the appropriate port number from the available alternatives.

Attribute	Description
User = *value*	The SQL Server login name to use when connecting to SQL Server. If the SQL Server instance uses Windows Authentication (also known as trusted connections), the Windows user name to use to authenticate the connection. If the SQL Server instance is running on a machine that is part of a Windows domain, use this format: *domain\username* If the SQL Server instance uses SQL Server Authentication, The SQL Server user name to use for the connection. To specify the login name in the connection string, use `UID` rather than `User`. For more information about specifying Easysoft ODBC-SQL Server Driver attributes in the connection string, see **"DSN-less Connections" on page 81**.
Password = *value*	The password for the login name specified by `User`. To specify the login password in the connection string, use `PWD` rather than `Password`.

Attribute	Description
Database = *value*	The default database to use for the connection. If you omit this attribute, the connection uses the default database defined for the login in SQL Server. The default database for users who do not have their own SQL Server login depends on the local group on the SQL Server machine that they belong to. The default database for members of the local Administrators group is the one defined for the BUILTIN\Administrators login. The default database for members of the local Users group is the one defined for the BUILTIN\Users login. If the database does not exist or the login does not have permission to access the database, the connection will fail. To check which databases you can access with your login, use `/usr/local/easysoft/sqlserver/bin/tdshelper`: `./tdshelper -s` *`instance`* `-n` *`port`* `-u` *`login`* `-a` *`password`* `-d` where *`instance`* is the SQL Server instance you specified with the `Server` attribute, *`port`* is the port number you specified with the `Port` attribute, *`login`* is the login name you specified with the `User` attribute and *`password`* is the password you specified with the `Password` attribute. The default database defined for your login is shown first in the `tdshelper` output. Note that you may have to enclose the instance name, login name and password with single quotes. Do this to protect any backslashes, spaces or other special characters these settings contain. For example, `'mydomain\myusername'`. Note that using the default database for the login ID is more efficient than specifying a default database in the ODBC data source.

Attribute	Description

QuotedId = 0 | 1	When ON (set to 1), `QUOTED_IDENTIFIERS` is set to ON for the connection. SQL Server will then follow the SQL-92 rules regarding the use of quotation marks in SQL Statements. Double quotes can only be used for identifiers, such as column and table names. Character strings must be enclosed in single quotes:
	`SELECT CompanyName`
	`FROM "Customer and Suppliers by City"`
	`WHERE City = 'New York'`
	If a single quotation mark is part of the literal string, it can be represented by two single quotation marks.
	When OFF, `QUOTED_IDENTIFIERS` is set to OFF for the connection. SQL Server then follows the legacy Transact-SQL rules regarding the use of quotation marks. Identifiers cannot be quoted and must follow all Transact-SQL rules for identifiers. Literals can be delimited by either single or double quotation marks.
	For more information about the `QUOTED_IDENTIFIERS` option, see the SQL Server Transact-SQL documentation.
	By default, `QuotedId` is ON.

Attribute	Description

AnsiNPW = 0 | 1	When ON (set to 1), the ANSI_NULLS, ANSI_WARNINGS, and ANSI_PADDING options are set to ON for the connection.
	When ANSI_NULLS is ON, SQL Server enforces ANSI rules for handling NULL comparisons. The ANSI syntax IS NULL or IS NOT NULL must be used for all NULL comparisons. For example:
	`SELECT *`
	`FROM MyTable`
	`WHERE MyColumn IS NULL`
	The Transact-SQL syntax = NULL and <> NULL are not supported.
	When ANSI_NULLS is OFF, the Equals (=) and Not Equal To (<>) comparison operators must be used to make comparisons with NULL and nonnull values in a table.
	When ANSI_WARNINGS is ON, SQL Server generates warning messages for conditions that violate ANSI rules but do not violate the rules of Transact-SQL. For example, SQL Server will generate error and warning messages for divide-by-zero errors, string too large for database column errors and when NULL values are encountered when using aggregate functions. When SET ANSI_WARNINGS is OFF, these errors and warnings are not raised.
	When ANSI_PADDING is ON, trailing blanks on varchar values and trailing zeroes on varbinary values are not automatically trimmed.
	For more information about the ANSI_NULLS, ANSI_WARNINGS, and ANSI_PADDING options, see the SQL Server Transact-SQL documentation.
	By default, AnsiNPW is ON.

Attribute	Description

Language = *value*	The national language to use for SQL Server system messages. Use this format:
	`Lanaguage = language`
	where `language` is one the language aliases contained in the `sys.syslanguages` table.
	For example:
	`Lanaguage = French`
	The computer running SQL Server must have the language installed. To check which languages are available, use `/usr/local/easysoft/sqlserver/bin/tdshelper`:
	`./tdshelper -s instance -n port -u login -a password -l`
	where *`instance`* is the SQL Server instance you specified with the `Server` attribute, *`port`* is the port number you specified with the `Port` attribute, *`login`* is the login name you specified with the `User` attribute and *`password`* is the password you specified with the `Password` attribute. The default language for your login is shown first in the `tdshelper` output.
	If no language is specified, the connection uses the default language specified for the login on the server.

Appname = *value*	The name SQL Server uses to identify the application that connects using this data source. For example, the following entry identifies an application as `isql`: `Appname = isql` The default value is `ODBC`. SQL Server stores the application name in the `master.dbo.sysprocesses` column `program_name`. The name is returned by the `APP_NAME` function.

Attribute	Description
MARS_Connection = 0 | 1	When ON (set to 1), multiple active result sets (MARS) are enabled on the connection if the server is SQL Server 2005. MARS allows applications to have more than one pending request per connection, and in particular, to have more than one active default result set per connection. Applications can execute other statements (for example, INSERT, UPDATE, DELETE, and stored procedure calls) while result sets are open. For example, an application might retrieve unprocessed items from an Orders table and then, while looping through the active result set, use an UPDATE statement to mark each order as processed. For non-MARS connections (`MARS_Connection` turned OFF) and earlier versions of SQL Server, applications cannot maintain multiple active statements on a connection. Applications that attempt to do this fail with the error "`connection is busy with results of another hstmt`". The application has to process or cancel all result sets from one batch before it can execute any other batch on that connection. Note that server-side cursors can be used to work around this limitation. There is a performance penalty associated with server-side cursors however. For more information about MARS, see the Microsoft article **Multiple Active Result Sets (MARS) in SQL Server 2005**. By default, `MARS_Connection` is OFF.
Wsid = *value*	The workstation ID. The default value is the host name of the machine where the ODBC application is running. SQL Server stores the workstation ID in the `master.dbo.sysprocesses` column `hostname`. The ID is returned by `sp_who` and the `HOST_NAME` function.

Attribute	Description
Version7 = 0 | 1	Set to `Version7` to 1 if you are connecting to a SQL Server 7.0 database. When initiating the connection, the Easysoft ODBC-SQL Server Driver tries to discover the version of the SQL Server instance. Setting `Version7` to 1 reduces the number of steps in the discovery process for SQL Server 7.0 databases. This results in a slightly quicker connection time. By default, `Version7` is OFF (set to 0).

Figure 3: Easysoft ODBC-SQL Server Driver data source settings.

ENVIRONMENT

The Easysoft ODBC-SQL Server Driver must be able to find the following shared objects, which are installed during the Easysoft ODBC-SQL Server Driver installation:

- `libodbcinst.so`

  By default, this is located in `/usr/local/easysoft/unixODBC/lib`.

- `libeslicshr.so`

  By default, this is located in `/usr/local/easysoft/lib`.

- `libessupp.so`

  By default, this is located in `/usr/local/easysoft/lib`.

- `libestdscrypt.so`

  By default, this is located in `/usr/local/easysoft/lib`.

  For more information about `libestdscrypt.so`, see **"Windows Authentication" on page 93**.

You may need to set and export `LD_LIBRARY_PATH`, `SHLIB_PATH` or `LIBPATH` (depending on your operating system and run-time linker) to include the directories where `libodbcinst.so`, `libeslicshr.so` and `libessupp.so` are located.

Note The shared object file extension (`.so`) may vary depending on the operating system (`.so`, `.a` or `.sl`).

ESTABLISHING A TEST CONNECTION

The `isql` query tool lets you test your Easysoft ODBC-SQL Server Driver data sources.

To test the Easysoft ODBC-SQL Server Driver connection

1. Change directory into `/usr/local/easysoft/unixODBC/bin`.
2. Type `./isql -v` *`data_source`*, where *data_source* is the name of the target data source.

   If you are unable to connect, see **"Troubleshooting Database Connection Problems" on page 75** for help on solving some common connection problems.
3. At the prompt, type an SQL query. For example:

   ```
   SQL> select * from mytable;
   ```

   – OR –

   Type `help` to return a list of tables:

   ```
   SQL> help
   ```

TROUBLESHOOTING DATABASE CONNECTION PROBLEMS

This section lists some common connection problems and their solutions.

- **Client unable to establish connection: OS Error: 'Failed to find host address 'myhost\myinstance'**
- **Client unable to establish connection: OS Error: 'Connection refused'**
- **Client unable to establish connection: Server not configured for TCP connection**
- **Client unable to establish connection: OS Error: 'Failed to get datagram from socket'**
- **Login failed for user ''. The user is not associated with a trusted connection**
- **Login failed for user 'myuser'.**

Client unable to establish connection: OS Error: 'Failed to find host address 'myhost\myinstance'

Check the Server attribute in your data source specifies a valid machine name or IP address. Check that the machine name can be looked up by using DNS or is present in `/etc/hosts`. Check that you are on the same network as the target host by pinging the machine:

```
ping myhost
```

If `ping` times out or fails, then either the DNS lookup is not working properly or there is some other networking or routing issue that needs to be resolved. Contact your network administrator.

Client unable to establish connection: OS Error: 'Connection refused'

Check that the SQL Server instance that you are trying to connect to is running.

If the SQL Server Browser or listener service is running, you can use `tdshelper` to display a list of all the running SQL Server instances on your network. On the machine where the Easysoft ODBC-SQL Server Driver is installed:

```
cd /usr/local/easysoft/sqlserver/bin
```

```
./tdshelper -i
```

On the SQL Server machine, "SQL Server <*instance*>" will be listed in output of the `net start` command, if the SQL Server instance is running.

If SQL Server is listening on a fixed TCP port, check that you can use telnet to connect to the port that you have specified in the data source:

```
telnet hostname port
```

where *hostname* is the host name or IP address of the machine where SQL Server is running and *port* is the port number that you have specified with the `Port` attribute. If the SQL Server instance is listening on this port, you will see output similar to:

```
Connected to myserver

Escape character is '^]'
```

To exit from telnet, type CTRL-] and then `quit`.

If you do not see this output or a "Connection refused" error displays, SQL Server is not listening on the specified port. Contact your database administrator for the correct SQL Server port.

If you are using the correct port but are unable to connect with telnet, the SQL Server instance may not allow remote TCP/IP connections. See **"Client unable to establish connection: Server not configured for TCP connection" on page 77**.

Client unable to establish connection: Server not configured for TCP connection

The TCP/IP protocol must be enabled in the instance that you are trying to connect to.

In the SQL Server Configuration Manager, in the list of network protocols for the instance, the status for TCP/IP must be set to "Enabled".

By default, SQL Server 2005 does not allow remote connections, which means that the default setting for TCP/IP is "Disabled".

Client unable to establish connection: OS Error: 'Failed to get datagram from socket'

The Easysoft ODBC-SQL Server Driver uses the SQL Server Browser or the SQL Server 2000 listener service to find out what TCP port SQL Server is listening on. If the SQL Server Browser or listener service is not running and active, the Easysoft ODBC-SQL Server Driver will be unable to open a connection for this purpose and the "`Failed to get datagram from socket`" error displays.

On the machine where the Easysoft ODBC-SQL Server Driver is installed, if you run `/usr/local/easysoft/tdshelper -i` and see no SQL Server instances listed in the output, the browser or listener may not be running.

On the SQL Server machine, "SQL Server Browser" will be listed in output of the `net start` command, if the SQL Server Browser is running. If `net start` shows that the SQL Server Browser service is running, the service may not be active. In the SQL Server Configuration Manager, the Active option must be set to "Yes" in the Advanced SQL Server Browser property tab. (The SQL Server Browser service must be restarted before any change to this setting takes effect.)

If you are connecting to SQL Server through a firewall, the firewall needs to allow connections through:

- The SQL Browser UDP port, 1434.
- The TCP port that the SQL Server instance is listening on.

If UDP port 1434 is not open, the firewall will block the connection when the Easysoft ODBC-SQL Server Driver attempts to discover the SQL Server port and the 'Failed to get datagram from socket' will display.

Because the SQL Server Browser or listener accepts unauthenticated UDP requests, it may have been turned off as a security measure, and your database administrator will have configured each SQL Server instance to listen on a specific TCP port. You need to specify this port number with the `Port` setting. For example, if SQL Server is listening on port 1500, add this line to the data source in `odbc.ini`:

```
Port = 1500
```

The "`Failed to get datagram from socket`" error also displays if you try to connect to a hidden SQL Server instance. You need to specify the port that the hidden instance is listening even though the SQL Server Browser or listener may be running.

Login failed for user ''. The user is not associated with a trusted connection

Check that the `User` and `Password` attributes for the data source in the `odbc.ini` specify a valid Windows user name and password.

This error also displays if you try to connect to SQL Server with a SQL Server user name and password but SQL Server's authentication mode is set to Windows Authentication only. To connect by using a SQL Server account, the security mode for the SQL Server instance must be changed to mixed (both SQL Server and Windows authentication are enabled).

To enable mixed mode, your database administrator must set the SQL Server security property **Server Authentication** to **SQL Server and Windows Authentication mode**. Note that Microsoft recommend that Windows authentication is used to connect to SQL Server whenever possible.

Login failed for user 'myuser'.

Check that the `User` and `Password` attributes for the data source in the `odbc.ini` specify a valid SQL Server user name and password.

This error also displays if you try to connect to SQL Server with a valid Windows user name and password but no corresponding SQL Server login exists. For example, SQL Server Setup creates a login named BUILTIN\Administrators that allows members of the local Administrators Windows group to access SQL Server. As a security measure, the database administrator may delete this login and members of this group will then need individual SQL Server login accounts to access SQL Server.

Ask your database administrator to create a SQL Server login for you that uses Windows authentication to validate your connection details.

DSN-less Connections

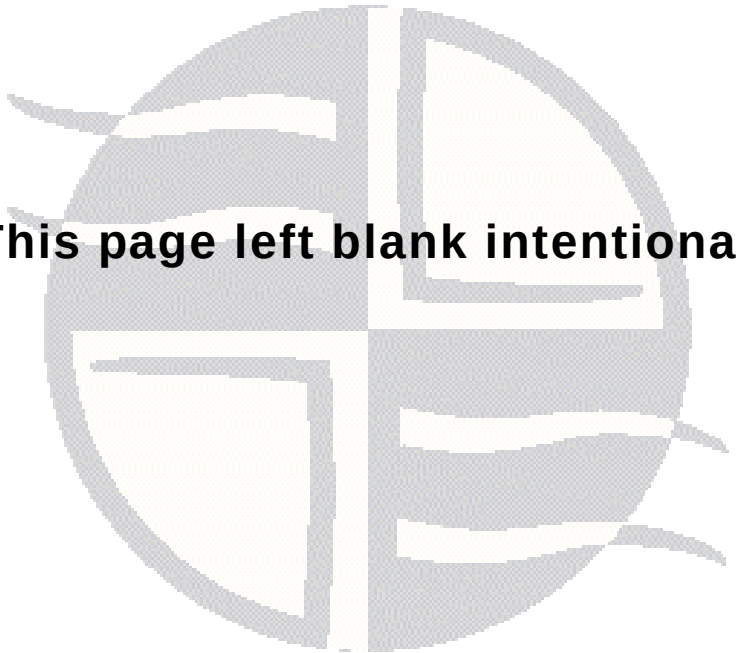
In addition to using a data source, you can also connect to a database by using a DSN-less connection string of the form:

```
SQLDriverConnect(..."DRIVER={Easysoft ODBC-SQL
Server};
                    Server=server;UID=user;PWD=password;
                    Port=port;"...)
```

where *server* is the SQL Server instance that you want to connect to, *user* and *password* are a valid SQL Server login and password and *port* is the TCP port that SQL Server is listening on. You need to use the `Easysoft ODBC-SQL Server` DRIVER keyword to identify the Easysoft ODBC-SQL Server Driver.

Other Easysoft ODBC-SQL Server Driver attribute settings, as described in **"Setting Up Data Sources" on page 61**, can be added to the connection string using the same `PARAMETER=value;` format. For example, the following connection string changes the default database with the `Database` attribute:

```
"DRIVER={Easysoft ODBC-SQL
Server};Server=myhost\\SQLEXPRESS;UID=mydomain\\myuser;PWD=mypassword;Port=1500
;Database=Sales;"
```



This page left blank intentionally

APPENDIX A TECHNICAL REFERENCE

Technical Reference for the Easysoft ODBC-SQL Server Driver

This section contains extra information relating to the deployment of the Easysoft ODBC-SQL Server Driver.

Appendix Guide

- **Restrictions**
- **ODBC Conformance**
- **The SQL Server 2005 xml Data Type**
- **Using Large-Value Data Types**
- **SQL Server Authentication Modes**
- **Encrypting Connections to SQL Server**
- **Threading**
- **Tracing**

Restrictions

The Easysoft ODBC-SQL Server Driver lets ODBC applications access:

- SQL Server 7.0
- SQL Server 2000
- SQL Server 2005
- SQL Server 2005 Express Edition

ODBC Conformance

The Easysoft ODBC-SQL Server Driver complies with the ODBC 3.51 specification.

- The driver conforms to all Core Level requirements.
- The driver conforms to 101, 102, 103, 104, 105, 106, 107, 108, 109 of the Level 1 requirements.
- The driver conforms to 201, 202, 203, 204, 205, 206, 207, 208, 209, 210 and 211 of the Level 2 requirements.

ODBC API Support

All ODBC 3.51 calls are supported.

Cursor Support

The Easysoft ODBC-SQL Server Driver supports FORWARD_ONLY, KEYSET_DRIVEN, DYNAMIC and STATIC cursors.

Supported Data Types

The Easysoft ODBC-SQL Server Driver supports the following SQL Server data types:

- `bigint`
- `binary`
- `bit`
- `char`
- `datetime`
- `decimal`
- `float`
- `image`
- `int`
- `money`
- `numeric`
- `real`
- `smalldatetime`
- `smallint`
- `smallmoney`
- `sql_variant`
- `text`
- `timestamp`
- `tinyint`
- `uniqueidentifier`

- varbinary
- varbinary(max)
- varchar
- varchar(max)
- xml

> **Note** varchar(max), nvarchar(max), varbinary(max) and xml are SQL Server 2005 data types.

SQLGetTypeInfo

SQL Server treats identity as an attribute, whereas ODBC treats it as a data type. To resolve this mismatch, SQLGetTypeInfo returns the data types: int identity, smallint identity, tinyint identity, decimal() identity, and numeric() identity. The SQLGetTypeInfo result set column AUTO_UNIQUE_VALUE reports the value TRUE for these data types.

For varchar, nvarchar and varbinary data types, the Easysoft ODBC-SQL Server Driver continues to report 8000, 4000 and 8000 for the COLUMN_SIZE value, even though it is actually unlimited. This is to ensure backward compatibility.

For the xml data type, the Easysoft ODBC-SQL Server Driver reports SQL_SS_LENGTH_UNLIMITED for COLUMN_SIZE to denote unlimited size.

Unicode Support

The Easysoft ODBC-SQL Server Driver is a Unicode driver that supports the Unicode version (with suffix "W") of the ODBC calls it implements. Using a Unicode driver with a Unicode application removes the need for the driver manager to map Unicode functions and data types to ANSI. This results in better performance and removes the restrictions inherent in the Unicode to ANSI mappings.

The Easysoft ODBC-SQL Server Driver supports the following SQL Server Unicode data types:

- `nchar`
- `ntext`
- `nvarchar`
- `nvarchar(max)`

Note `nvarchar(max)` is a SQL Server 2005 data type.

The SQL Server 2005 xml Data Type

SQL Server 2005 provides an `xml` data type for storing XML documents in table columns or Transact-SQL variables.

The Easysoft ODBC-SQL Server Driver supports the SQL Server 2005 `xml` data type and its associated methods: `query()`, `value()`, `exist()`, `modify()` and `nodes()`.

The `query()` method lets you use an XML Query (XQuery) definition to search XML data stored in columns and variables of the `xml` type. The XQuery language is a World Wide Web Consortium (W3C) standard for retrieving or defining a set of XML nodes that meet a set of criteria.

In the following example, an XQuery is specified against the Instructions column in the ProductModel table. The Instructions column data type is `xml` and therefore exposes the `query()` method. The ProductModel table is contained in the SQL Server sample database AdventureWorks.

```
SELECT Instructions.query('declare namespace
AWMI="http://schemas.microsoft.com/sqlserver/2004/07/adventure-
works/ProductModelManuInstructions";
    /AWMI:root/AWMI:Location[@LocationID=10]
') as Result
FROM  Production.ProductModel
WHERE ProductModelID=7
```

The XQuery includes a namespace declaration, `declare namespace AWMI=...`, and a query expression, `/AWMI:root/AWMI:Location[@LocationID=10]`. The namespace declaration identifies the XML namespace associated with elements in the Instructions column. The query expression retrieves only those records for which the `LocationID` attribute value is 10:

```
<AWMI:Location
xmlns:AWMI="http://schemas.microsoft.com/sqlserver/2004/07/adventu
re-works/ProductModelManuInstructions"
LaborHours="2.5"...LocationID="10">

...
```

This second example uses the `query()` method to construct an XML element named `<Product>`. The `<Product>` element has a `ProductModelID` attribute, in which the `ProductModelID` attribute value is retrieved from the database.

```
SELECT CatalogDescription.query(' declare namespace
PD="http://schemas.microsoft.com/sqlserver/2004/07/adventure-
works/ProductModelDescription"; <Product ProductModelID="{
/PD:ProductDescription[1]/@ProductModelID }" /> ') as Result

FROM Production.ProductModel
```

The `exist()` method lets you filter XML data. For example, add the following WHERE clause to the previous query to find only records that contain a <`Warranty`> element.

```
where CatalogDescription.exist(' declare namespace
PD="http://schemas.microsoft.com/sqlserver/2004/07/adventure-
works/ProductModelDescription"; declare namespace
wm="http://schemas.microsoft.com/sqlserver/2004/07/adventure-
works/ProductModelWarrAndMain";
/PD:ProductDescription/PD:Features/wm:Warranty ') = 1
```

Note

When querying or updating `xml` columns or variables with the `xml` data type methods, the data source attributes `AnsiNPW and QuotedId` must be set to 1 (the default value for these settings). Otherwise, queries and modifications will fail for `xml` data types.

Using Large-Value Data Types

Microsoft SQL Server 2005 provides the max specifier, which expands the storage capabilities of the `varchar`, `nvarchar`, and `varbinary` data types to allow storage of values as large as 2 gigabytes (GB). `varchar(max)`, `nvarchar(max)`, and `varbinary(max)` are collectively called large-value data types.

The Easysoft ODBC-SQL Server Driver exposes the `varchar(max)`, `varbinary(max)` and `nvarchar(max)` types as `SQL_VARCHAR`, `SQL_VARBINARY`, and `SQL_WVARCHAR` in ODBC API functions that accept or return ODBC SQL data types.

When reporting the maximum size of a column, the Easysoft ODBC-SQL Server Driver will report either:

- The defined maximum size, which for example, is 2000 for a `varchar(2000)` column.

– OR –

- The value `SQL_SS_LENGTH_UNLIMITED` (0) `for varchar(max)`, `varbinary(max)` and `nvarchar(max)` columns.

SQL Server Authentication Modes

Users are granted access to SQL Server instances through a SQL Server login. SQL Server provides two ways to authenticate SQL Server logins: Windows Authentication (also known as trusted connections) and SQL Server Authentication.

Windows Authentication allows users to connect to SQL Server by using their Windows user account. SQL Server uses the Windows security system to validate these trusted connections.

SQL Server authentication uses passwords stored in SQL Server to validate the connection.

Windows Authentication is Microsoft's recommended SQL Server authentication mode because it provides the following advantages:

- User names and passwords are encrypted.
- Security is easier to manage (a single Windows security model instead of a separate SQL Server security model).
- Login security improves through password expiration, minimum password lengths, and account lockout policies.

The Easysoft ODBC-SQL Server Driver supports both Windows Authentication and SQL Server Authentication. You specify the SQL Server login name and password with the `User` and `Password` data source attributes or the `UID` and `PWD` connection string attributes.

Windows Authentication

If the `User` attribute value contains a backslash (used to separate a user name from a domain), for example, `mydomain\myuser`, the Easysoft ODBC-SQL Server Driver asks SQL Server to use Windows Authentication to validate the connection. The Easysoft ODBC-SQL Server Driver passes the domain user name and password to SQL Server in an encrypted form. This process involves both the Data Encryption Standard (DES) encryption method and the MD4 hashing algorithm. The Easysoft ODBC-SQL Server Driver uses open source code for both these methods. The code is distributed under the terms of the GNU Lesser General Public License (LGPL). To read the license, see `/usr/local/easysoft/sqlserver/crypt/COPYING`.

To comply with the terms of the LGPL, the encryption functions are not included in the main Easysoft ODBC-SQL Server Driver library. Instead, they are provided in the shared library file `/usr/local/easysoft/lib/libestdscrypt.so`. The Easysoft ODBC-SQL Server Driver distribution includes the source files for this library. The source files are installed in `/usr/local/easysoft/sqlserver/crypt`. The supplied Makefile will build the library on your Easysoft ODBC-SQL Server Driver platform.

SQL Server Authentication

If the `User` attribute value does not contain a backslash, the Easysoft ODBC-SQL Server Driver asks SQL Server to use SQL Server Authentication to validate the connection. The Easysoft ODBC-SQL Server Driver sends the password to SQL Server in an encrypted form, although the encryption is less strong then that used for trusted connections. The SQL Server user name is sent in plain text.

Encrypting Connections to SQL Server

SQL Server 2000 and SQL Server 2005 can use Secure Sockets Layer (SSL) to encrypt data transmitted across a network between an instance of SQL Server and a client application.

The Easysoft ODBC-SQL Server Driver with SSL Support lets Linux and Unix applications access SQL Server 2000 and 2005 over an encrypted connection. The SSL version of driver is included in the Easysoft ODBC-SQL Server Driver distribution and should be used instead of the standard Easysoft SQL Server driver whenever an SSL connection is required.

Installing an SSL Certificate

Before you can access SQL Server 2000 over an encrypted connection, an SSL certificate needs to be provisioned on the SQL Server machine. SQL Server 2005 can use an SSL certificate from a trusted CA if available or generate a self-signed certificate.

Self-Signed SSL Certificates

SQL Server 2005 will always make encryption available, even if a SSL certificate has not been installed on the server machine. For this reason, SQL Server 2005 always ensures that network packets associated with logging in are encrypted. If no certificate has been provisioned on the server when it starts up, SQL Server generates a self-signed certificate which is used to encrypt login packets.

Note	Even though SQL 2005 can make encryption available without an SSL certificate, Microsoft recommend using a certificate signed by a trusted authority whenever possible. SSL connections that are encrypted with a self-signed certificate protect against packet sniffing but do not protect against man-in-the-middle attacks. In a man-in the-middle attack, attackers route packets through their servers, which sniff the contents as they pass through.

Installation Information

If an SSL certificate has not yet been installed on the SQL Server machine, refer to the following Microsoft documentation for installation information:

- SSL certificate requirements for SQL Server 2000:

  http://support.microsoft.com/kb/318605

- SSL certificate requirements for SQL Server 2005:

  http://blogs.msdn.com/sql_protocols/archive/2005/12/30/508311.aspx

- Installing an SSL certificate on an SQL Server 2000 machine:

  http://support.microsoft.com/kb/316898

- Installing an SSL certificate on an SQL Server 2005 machine:

  http://msdn.microsoft.com/en-us/library/ms189067.aspx

TESTING THAT SSL IS AVAILABLE ON THE SQL SERVER

1. Do one of the following:
   - For SQL Server 2005, in SQL Server Configuration Manager, double-click SQL Server 2005 Network to expand the Protocols list. Right-click Protocols for the instance that you want to connect to and click Properties. Make sure that `ForceEncryption` is set to `Yes`.
   - For SQL Server 2000, in the SQL Server Network Utility, make sure that the `Force protocol encryption` option is checked.
2. Restart the instance.

3. Check the SQL Server error log (`Program Files\Microsoft SQL Server\MSSQL.`*n*`\MSSQL\LOG\ERRORLOG`, by default) to verify that SQL Server did not report any errors when it started.

> You can verify that SQL Server 2005 has successfully generated a self-signed SSL certificate by checking the SQL Server error log for a line containing:
>
> ```
> A self-generated certificate was successfully loaded for encryption.
> ```
>
> **Note** If SQL Server is unable to generate a self-signed certificate, you will be unable to connect to the instance over an encrypted connection. Note that when testing SSL with SQL Server Express, we had to change the account used by the SQL Server instance from `Network Service` to `Local System` before the instance could generate a certificate.

INSTALLING THE ROOT CA CERTIFICATE

If encryption is enabled on the SQL Server machine, the client machine does not have to trust the CA that signed the SSL certificate used by the instance, and you can skip this section.

If you request encryption from the client machine **rather than** the SQL Server machine, the Easysoft ODBC-SQL Server Driver with SSL Support must be able to verify the ownership of the certificate used by the SQL Server instance. (Unless the SQL Server 2005 instance is using a self-generated certificate, in which case you should use `TrustServerCertificate` attribute to bypass the verification process.) If the server certificate was signed by a public or private certificate authority (CA) for which the client machine does not have the public key certificate, you must install the public key certificate of the CA that signed the server certificate. If the client machine already has the public key certificate of the CA that signed the server certificate, this step is not necessary.

1. On the SQL Server machine, in the Windows Run dialog box, type:

   ```
   mmc
   ```

2. In Microsoft Management Console, on the File Menu, click Add/Remove Snap-in, and then click Add.
3. In the Add Standalone Snap-In dialog box, double-click Certificates. Click Computer account when prompted and then click Next. Click Finish.
4. Click Close and then OK to close the Add/Remove Snap-in dialog boxes.
5. In the Certificates Snap-in, locate the certificate for the CA that signed the SQL Server certificate. For example, if the CA certificate is in the Trusted Root Certificate Authorities store, double-click Trusted Root Certificate Authorities and click Certificates. In the right pane of the console window, right-click the CA certificate, point to All Tasks, and then click Export.

6. Complete the Certificate Export wizard.

   When prompted to choose the export format, make sure that you choose **Base-64 encoded X.509**.

7. Use FTP to copy the exported CA certificate to the client machine from which you want to access SQL Server.

CONFIGURING THE CLIENT TO REQUEST AN ENCRYPTED CONNECTION

SSL encryption can be enabled either on the SQL Server machine or the client machine. If encryption is enabled on the SQL Server machine, all connections to the machine are encrypted. If you do not want to enable encryption globally on the SQL Server machine, you can configure the client machine to request an SSL connection. The following steps show how to configure an Easysoft ODBC-SQL Server Driver with SSL Support data source that requests an encrypted connection.

1. Do one of the following:

   - For SQL Server 2005, in SQL Server Configuration Manager, double-click SQL Server 2005 Network to expand the Protocols list. Right-click Protocols for the instance that you want to connect to and click Properties. Make sure that `ForceEncryption` is set to `No`.
   - For SQL Server 2000, in the SQL Server Network Utility, make sure that the `Force protocol encryption` option is clear.

2. On the machine where the Easysoft ODBC-SQL Server Driver with SSL Support is installed, create an ODBC data source to connect to the SQL Server instance.

   In your data source, the `Driver` attribute must be set to `Easysoft ODBC-SQL Server SSL`.

Set the `Encrypt` and `TrustServerCertificate` attributes to `No`. For example:

```
[SQLSERVER_SSL_CONNECTION]
Driver      = Easysoft ODBC-SQL Server SSL
Server      = MYSQLSERVERMACHINE\MYINSTANCE
User        = MYDOMAIN\myuser
Password    = mypassword
Encrypt     = No
TrustServerCertificate= No
```

3. Use an application such as Microsoft Network Monitor, Snort or Ethereal to capture network traffic between this machine and the SQL Server machine.

   Using Network Monitor or a network sniffer tool lets you verify that you have successfully made an encrypted connection to the SQL Server machine. When testing the Easysoft driver, we used Network Monitor on the SQL Server machine and Snort on the client machine to capture network traffic.

   For information about using Network Monitor, see the Microsoft Knowledge Base article How to capture network traffic with Network Monitor **(http://support.microsoft.com/kb/148942/EN-US/**).

4. Use `isql` to connect to the data source and retrieve some data. For example:

```
$ cd /usr/local/easysoft/unixodbc/bin
$ ./isql -v SQLSERVER_SSL_CONNECTION
SQL> select * from HumanResources.EmployeePayHistory where EmployeeID = 1
```

5. In your network sniffer, verify that the data returned by the Easysoft ODBC-SQL Server Driver with SSL Support is not encrypted.

   This fragment of example output shows unencrypted data captured on the client machine by running `snort -vde`:

   ```
   8.E.m.p.l.o.y.e
   e.I.D.......=.R
   a.t.e.C.h.a.n.g
   e.D.a.t.e......
   <.R.a.t.e......
   0.P.a.y.F.r.e.q
   u.e.n.c.y......
   =.M.o.d.i.f.i.e
   d.D.a.t.e......
   ```

6. Press RETURN to exit `isql`.
7. In your data source, set the `Encrypt` attribute to `Yes`.

8. Do one of the following:

   - To connect to a SQL Server 2000 instance or SQL Server 2005 instance on a machine where an SSL certificate has been provisioned, use the `CertificateFile` attribute to specify the path to the CA certificate file. The certificate file must contain the public key certificate of the CA that signed the SQL Server certificate. The public key certificate must be in base-64 PEM format.

     If the CA's public key was already installed on this machine, specify the path to the CA store that contains the public key. For example, `CertificateFile = /usr/share/ssl/certs/ca-bundle.crt`. If you exported the CA certificate on the SQL Server machine and copied it to this machine, specify the path to the certificate file. For example, `CertificateFile = /usr/share/ssl/CA/MyCA.cer`. For more information, see **"Installing the Root CA Certificate" on page 97**.

   - To connect to a SQL Server 2005 instance that is using a self-signed certificate, set the `TrustServerCertificate` attributes to `Yes`.

9. Use `isql` to connect to the data source and retrieve the same data as you did in step 4.

10. In your network sniffer, verify that the data returned by the Easysoft ODBC-SQL Server Driver with SSL Support is now encrypted.

    This example output shows encrypted SQL Server data captured in Snort.

```
.E..{i.2.8.G.q..
.n..{X.... 4..O.
&....Lt..Z.wrH.8
.W..{..........,
....1s_..).\k.6.
..4U..4..D...5.U
&...I......+..w.
l.W...&}x.......
....%......7...J
..C$...,j..52.~.
.w. Q.qE.Q....]4
.\.Y?...|R.VOr.S
.....K.W.. 2.#.T
.G..+..F.....T..
@"..+-.......
```

EASYSOFT ODBC-SQL SERVER DRIVER WITH SSL SUPPORT ATTRIBUTE FIELDS

The following attributes may be set in the `odbc.ini` file:

Attribute	Description
Encrypt = Yes | No	Whether the client requests an encrypted connection to SQL Server. If you do not want to enable SSL encryption globally at the server, you can enable SSL encryption on a per client basis. To do this, set `Encrypt` to `Yes`. Do not enable SSL encryption on both the server and client, use one or the other.

Attribute	Description
TrustServerCertificate = Yes | No	Enables the client to request encryption even when an SSL certificate has not been provisioned on the SQL Server 2005 machine. If SQL Server 2005 cannot load a valid SSL certificate at startup time, it will generate a self-signed certificate to make encryption available. When the client requests encryption by setting `Encrypt` to Yes, the Easysoft ODBC-SQL Server Driver with SSL Support tries to validate the server certificate to verify the identity of the server machine. This is impossible to do with a self-signed certificate since it has not been signed by a trusted root authority. Setting `TrustServerCertificate` to Yes overrides the server validation. If the SQL Server 2005 `ForceEncryption` option is enabled, the `TrustServerCertificate` value is ignored. When encryption is enabled on the SQL Server machine, the Easysoft ODBC-SQL Server Driver with SSL Support bypasses the validation of the server certificate. Note that SQL Server 2000 cannot generate a self-signed certificate. SSL encryption is only available if the SQL Server 2000 instance is running on a computer that has a certificate assigned from a public certification authority. By default, `TrustServerCertificate` is ON (set to `Yes`).

Attribute	Description

CertificateFile = *filename*	The file that contains the public key certificate of the CA that signed the SQL Server certificate. The CA certificate file must be in base-64 PEM format.
	If the CA certificate is not installed on your client machine, you need to export the certificate on the SQL Server machine and install it on the client. For more information, see **"Installing the Root CA Certificate" on page 97**.
	Examples
	To load a CA certificate from the root CA certificate store supplied with the OpenSSL distribution, use:
	`CertificateFile = /usr/share/ssl/certs/ca-bundle.crt`
	To load a private CA certificate named `MyCA.cer` that you copied to `/usr/share/ssl/CA`, use:
	`CertificateFile = /usr/share/ssl/CA/MyCA.cer`

Entropy = *filename*	OpenSSL needs a source of unpredictable data to work correctly. Many open source operating systems provide a "randomness device" (`/dev/urandom` or `/dev/random`) that serves this purpose. The Easysoft ODBC-SQL Server Driver with SSL Support tries to use `/dev/urandom` by default and will also try to use `/dev/random` if `/dev/urandom` is not available. On systems without `/dev/urandom` or `/dev/random`, the EGD entropy gathering daemon can be used as an alternative source of random data. It provides a socket interface through which entropy (randomness) can be gathered. Use the `Entropy` attribute to specify the path to the EGD socket. For example: `Entropy = /dev/egd-pool`

Figure 4: Easysoft ODBC-SQL Server Driver with SSL Support data source settings.

Threading

The Easysoft ODBC-SQL Server Driver is thread safe in accordance with the ODBC 3.51 specification and can safely be used behind threaded applications.

Tracing

The ODBC calls an application makes can be traced:

- Within the Driver Manager by an application.
- From within the Driver Manager.
- From within the Easysoft ODBC-SQL Server Driver.

WITHIN THE DRIVER MANAGER BY AN APPLICATION

An application can turn tracing on in the Driver Manager by using the ODBC API `SQLSetConnectAttr` (...,`SQL_ATTR_TRACE`,...).

The trace file name may also be specified with the `SQLSetConnectAttr` attribute `SQL_ATTR_TRACEFILE`.

FROM WITHIN THE DRIVER MANAGER

For the unixODBC Driver Manager, add two attributes to the [`ODBC`] section (create one if none exists) in `odbcinst.ini`.

```
Trace = Yes
TraceFile = /path/filename
```

For example:

```
[ODBC]
Trace = Yes
TraceFile = /tmp/sql.log
```

Ensure that the user who is running the application to be traced has write permission to `TraceFile` (and to the directory containing it), or no tracing information will be produced.

From within the Easysoft ODBC-SQL Server Driver

Add a LOG attribute to the DSN section of the odbc.ini file.

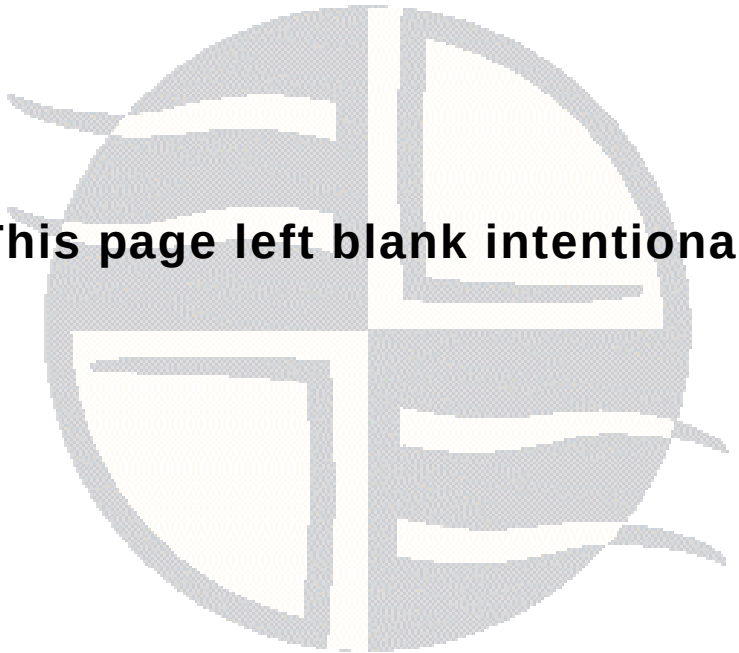
For example:

```
[SQLSERVER]

....

LOG = /tmp/sql.log
```

The LOG value is the path and file name of the log file, for example, /tmp/sql.log.



This page left blank intentionally

APPENDIX B GLOSSARY

Terms and definitions

Application Programmer Interface (API)

A published set of function calls and constants allowing different programmers to utilize a ready-written library of subroutines.

Authorization code

You must have an authorization code for the Easysoft product you wish to license in order to obtain a purchased license. When you purchase a product your authorization code is emailed to you. You do not need an authorization code to obtain a trial license.

Batch

A set of SQL statements submitted together and run as a group. A script is often a series of batches submitted one after the other.

Binary large object

A piece of binary data that has an exceptionally large size, such as pictures or audio tracks that are stored as digital data, or any variable or table column that is large enough to hold such values. In Transact-SQL, a binary large object (BLOB) is stored in an image column. Sometimes the term BLOB is also applied to large character data values, such as those stored in text or ntext columns.

Client

A process performing tasks local to the current user, for example, formatting and displaying a report from data retrieved from the server.

Column

In an SQL table, the area in each row that stores the data value for some attribute of the object modeled by the table. For example, the Employee table in the AdventureWorks sample database models the employees of the Adventure Works Cycles company. The Title column in each row of the Employee table stores the job title of the employee represented by that row, the same way a Job Title field in a window or form would contain a job title.

Commit

An operation that saves all changes to databases, cubes, or dimensions made since the start of a transaction. A commit guarantees that all of the transaction's modifications are made a permanent part of the database, cube or dimension. A commit also frees resources, such as locks, used by the transaction.

Concurrency

A process that allows multiple users to access and change shared data at the same time. SQL Server uses locking to allow multiple users to access and change shared data at the same time without conflicting with each other.

Constraint

A property assigned to a table column that prevents certain types of invalid data values from being placed in the column. For example, a `UNIQUE` or `PRIMARY KEY` constraint prevents you from inserting a value that is a duplicate of an existing value, a `CHECK` constraint prevents you from inserting a value that does not match a specified condition, and `NOT NULL` prevents you from inserting a `NULL` value.

Cursor

An entity that maps over a result set and establishes a position on a single row within the result set. After the cursor is positioned on a row, operations can be performed on that row, or on a block of rows starting at that position. The most common operation is to fetch (retrieve) the current row or block of rows.

Data Encryption Standard (DES)

A specification for encryption of computer data developed by IBM. DES uses a 56-bit key.

Data Definition Language

The subset of SQL statements that define all attributes and properties of a database and its objects. DDL statements typically start with CREATE, ALTER, or DROP.

Data Manipulation Language

The subset of SQL statements that is used to retrieve and manipulate data. DML statements typically start with SELECT, INSERT, UPDATE, or DELETE.

Data source

In ODBC terms, a data source is a database or other data repository coupled with an ODBC Driver, which has been given a Data Source Name (see **"DSN" on page 116**) to identify it to the ODBC Driver Manager.

Data type

An attribute that specifies what type of information can be stored in a column, parameter, or variable. System-supplied data types are provided by SQL Server. Alias data types can also be created.

Database mirroring

The process of immediately reproducing every update to a read/write database (the principal database) onto a read-only mirror of that database (the mirror database) that resides on a separate instance of the Database Engine (the mirror server).

Deadlock

A situation when two users, each having a lock on one piece of data, attempt to acquire a lock on the other's piece. Each user would wait indefinitely for the other to release the lock, unless one of the user processes is terminated. SQL Server detects deadlocks and terminates one user's process.

Default database

The database the user is connected to immediately after logging in to SQL Server.

Default instance

The instance of SQL Server that uses the same name as the computer name on which it is installed.

Default language

The language that SQL Server uses for errors and messages if a user does not specify a language. Each SQL Server login has a default language.

Default result set

The default mode that SQL Server uses to return a result set back to a client. Rows are sent to the client in the order in which they are placed in the result set, and the application must process the rows in this order. After running an SQL statement on a connection, the application cannot do anything on the connection except retrieve the rows in the result set until all the rows have been retrieved. The only other action that an application can perform before the end of the result set is to cancel the remainder of the result set. This is the fastest method to get rows from SQL Server to the client.

Distributed query

A single query that accesses data from multiple data sources.

Distributed transaction

A transaction that spans multiple data sources. In a distributed transaction, all data modifications in all accessed data sources are either committed or terminated.

DBMS

Database Management System -- software that handles access to a database.

Driver

See **"ODBC driver" on page 119**.

Driver Manager

Software whose main function is to load ODBC drivers. ODBC applications connect to the Driver Manager and request a data source name (DSN). The Driver Manager loads the driver specified in the DSN's configuration file. On Windows, the ODBC Data Source Administrator is used to set up the Driver Manager.

DSN

Data Source Name. A name associated with an ODBC data source. Driver Managers, such as unixODBC or the Microsoft Windows Driver Manager, use the Data Source Name to cross-reference configuration information and load the required driver.

DSN-less connection

A type of data connection that is created based on information in a data source name (DSN), but is stored as part of a project or application. DSN-less connections are especially useful for Web applications because they let you move the application from one server to another without re-creating the DSN on the new server.

Field

A placeholder for a single datum in a record, for example you can have a Surname field in a Contact Details record. Fields are sometimes referred to as cells.

File

In SQL Server databases, a basic unit of storage for a database. One database can be stored in several files. SQL Server uses three types of files: data files (which store data), log files (which store transaction logs), and backup files (which store backups of a database).

Host

A computer visible on the network.

Identity column

A column in a table that has been assigned the identity property. The identity property generates unique numbers.

Index

In a relational database, a database object that provides fast access to data in the rows of a table, based on key values. Indexes can also enforce uniqueness on the rows in a table. SQL Server supports clustered and nonclustered indexes. The primary key of a table is automatically indexed. In full-text search, a full-text index stores information about significant words and their location within a given column.

Instance

A copy of SQL Server running on a computer. A computer can run multiple instances of SQL Server 2005. A computer can run only one instance of SQL Server version 7.0 or earlier, although in some cases it can also be running multiple instances of SQL Server 2000.

Integer

A data type category that includes the `bigint`, `int`, `smallint`, and `tinyint` data types.

Isolation level

The property of a transaction that controls the degree to which data is isolated for use by one process, and is guarded against interference from other processes. Setting the isolation level defines the default locking behaviour for all SELECT statements in your SQL Server session.

License key

A string that is provided by Easysoft for use in the licensing process.

Master database

The system database that records all the system-level information for an instance of SQL Server. This includes instance-wide metadata such as login accounts, endpoints, linked servers, and system configuration settings. Also, master is the database that records the existence of all other databases and the location of those database files and records the initialization information for SQL Server.

MD4

A hashing algorithm that creates a 128-bit hash value used to verify data integrity. A hashing algorithm is a mathematical procedure for randomising information to make it more secure in transmission. The more bits in a hash, the greater the security of the encryption process.

Multiple instances

Multiple copies of SQL Server running on the same computer. There can be one default instance, which can be any version of SQL Server. There can be multiple named instances of SQL Server 2000 and SQL Server 2005.

Named instance

An installation of SQL Server that is given a name to differentiate it from other named instances and from the default instance on the same computer. A named instance is identified by the computer name and instance name.

NULL

An entry that has no explicitly assigned value. `NULL` is not equivalent to zero or blank. A value of `NULL` is not considered to be greater than, less than, or equivalent to any other value, including another value of `NULL`.

ODBC

Open Database Connectivity -- a programming interface that enables applications to access data in database management systems that use Structured Query Language (SQL) as a data access standard.

ODBC driver

Software that accesses a proprietary data source, providing a standardized view of the data to ODBC.

Query optimizer

The SQL Server Database Engine component responsible for generating efficient execution plans for SQL statements.

Record

A group of related fields (columns) of information treated as a unit. A record is more commonly called a row in a relational database.

Result set

The set of rows returned from a SELECT statement. The format of the rows in the result set is defined by the column-list of the SELECT statement.

Row

In an SQL table, a single occurrence of the object modeled by the table. For example, in the AdventureWorks sample database, the Employee table models the employees of the Adventure Works Cycles company. Each row in the table records all the information about a specific employee such as an employee identification number, job title, and the date that employee was hired.

Server cursor

A cursor implemented on the server. The cursor itself is built at the server, and only the rows fetched by an application are sent to the client.

Server name

A name that uniquely identifies a server computer on a network. SQL Server applications can connect to a default instance of SQL Server by specifying only the server name. SQL Server applications must specify both the server name and instance name when connecting to a named instance on a server.

Structured Query Language (SQL)

A language used to insert, retrieve, modify, and delete data in a relational database, designed specifically for database queries. SQL also contains statements for defining and administering the objects in a database. SQL is the language supported by most relational databases, and is the subject of standards published by the International Standards Organization (ISO) and the American National Standards Institute (ANSI). SQL Server 2005 uses a version of the SQL language called Transact-SQL.

SQL-92

The version of the SQL standard published in 1992. The international standard is ISO/IEC 9075:1992 Database Language SQL. The American National Standards Institute (ANSI) also published a corresponding standard (Data Language SQL X3.135-1192), so SQL-92 is sometimes referred to as ANSI SQL in the United States.

Stored procedure

A precompiled collection of Transact-SQL statements that are stored under a name and processed as a unit. SQL Server supplies stored procedures for managing SQL Server and displaying information about databases and users. SQL Server-supplied stored procedures are called system stored procedures.

System databases

A set of five databases present in all instances of SQL Server that are used to store system information. The msdb database is used by SQL Server Agent to record information on jobs, alerts, and backup histories. The model database is used as a template for creating all user databases. The tempdb database stores transient objects that only exist for the length of a single statement or connection, such as worktables and temporary tables or stored procedures. The master database stores all instance-level metadata, and records the location of all other databases. The Resource database contains all the system objects that are included with SQL Server, such as system stored procedures and system tables.

Table

A data set in a relational database, composed of rows and columns.

Tabular Data Stream (TDS)

The SQL Server internal client/server data transfer protocol. TDS allows client and server products to communicate regardless of operating-system platform, server release, or network transport.

tempdb database

The database that provides a storage area for temporary tables, temporary stored procedures, and other temporary working storage needs.

Transaction

A group of database operations combined into a logical unit of work that is either wholly committed or rolled back. A transaction is atomic, consistent, isolated, and durable.

INDEX

A

AnsiNPW
    xml data type methods and 90
API conformance 84
attributes
    for connection strings 62
    for Easysoft ODBC-SQL Server Driver data sources 62

C

connection
    DSN-less 81
    testing 74
connection problems, troubleshooting 75
connection string 81
connection string attributes
    Password 63
    Port 65
    Server 63
    User 63
    Version7 72

D

data source attributes
    Description 64
    Driver 62
    Password 63
    Port 65
    Server 63
    User 63
    Version7 72
data sources 61-72
    adding 61
    connecting to 74
    example 24

definitions 111
Description 64
Driver 62
Driver Manager 21
DSN-less connections 81
dynamic linker search path 45

E

Easysoft FTP site 16
Easysoft ODBC-SQL Server Driver
    adding data sources 61
    downloading 16
    installing 19
    licensing 34
    setting the environment 72
    upgrading 18
encrypting the SQL Server connection 94
environment 72
environment variables
    LD_LIBRARY_PATH 46
    LD_RUN_PATH 46
    LIBPATH 46
EULA 29

F

Failed to get datagram from socket 78
files
    license.txt 29
    license_request.txt 36
    odbc.ini 61
    odbcinst.ini 62
firewalls, connecting to SQL Server through 78

I

installation
- changes made to your system 23
- default installation path 22
- non-root installations 21
- other Easysoft products and 27
- running 30
- system requirements 19
- unixODBC and 31
- unpacking the distribution 28
- what you need to know 28

installing the Easysoft ODBC-SQL Server Driver 19
isql 74

L

LD_LIBRARY_PATH 45, 46, 73
LD_RUN_PATH 45, 46
LIBPATH 45, 46, 73
license.txt 29
license_request.txt 37
licenses.out 37
licensing
- Easysoft ODBC-SQL Server Driver and 34
- End-User License Agreement 29

N

non-root installations 43

O

ODBC
- conformance 84
- tracing 108
- unixODBC Driver Manager 21

odbc.ini .....61-72, ??-106
Description attribute 64
Driver attribute 62
password attribute 63
Port attribute 64
Server attribute 63
user attribute 63

P

Password 63
Port 64

Q

QuotedId
xml data type methods and 90

S

Server 63
SHLIB_PATH 45, 46, 73
SSL encryption 94

T

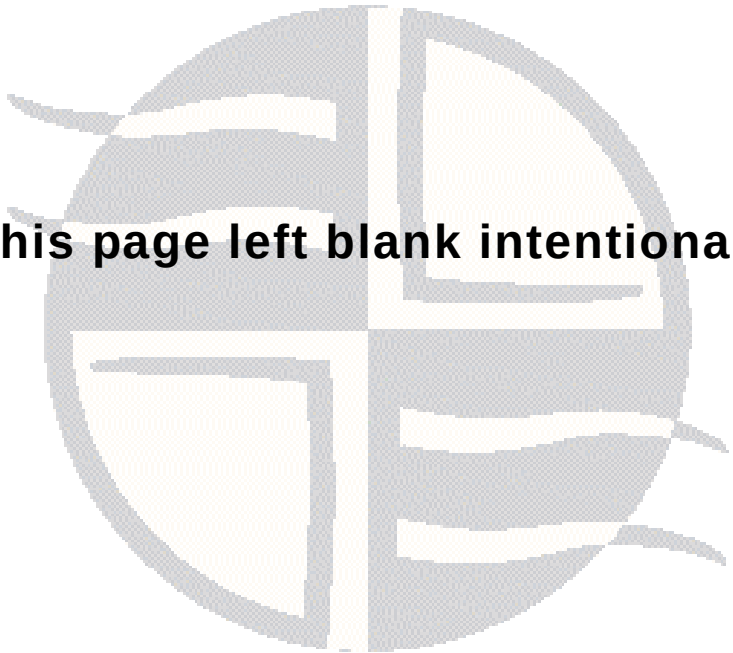
Tabular Data Stream 121
tdshelper 37
thread-safety 107
tracing ODBC API calls 108
troubleshooting 75
connection refused errors 76
failed to find host address errors 76
failed to get datagram from socket errors 78
server not configured for TCP connection errors 77
trusted connections 92

U

uninstalling 48
unixODBC 31
    configure options 32
    Easysoft ODBC-SQL Server Driver and 31
unpacking the Easysoft ODBC-SQL Server Driver distribution 28
User 63

W

Windows Authentication 92
Windows NT Authentication 92



This page left blank intentionally.