

Easysoft ODBC- Sybase Driver User's Guide

This manual documents version 1.3.n of the Easysoft ODBC-Sybase Driver.

Copyright © 1993-2025 Easysoft Limited.

All rights reserved.

You may not reverse engineer, decompile, or disassemble this manual. Information in this document is subject to change without notice. Companies, names, and data used in examples are fictitious unless otherwise noted.

The names of companies referred to herein, their corporate logos, the names of their hardware and software may be trade names, trademarks or registered trademarks of their respective owners.

Easysoft and the Easysoft logo are registered trademarks of Easysoft Limited.

The software described in this document is provided under a [licence agreement](#) and may be used only in accordance with the terms of that agreement.

Table of contents

Getting started	4
Installing the Easysoft ODBC-Sybase Driver	5
Installing on Linux or UNIX	5
Uninstalling on Linux or UNIX	13
Connecting to Sybase	14
Connecting from Linux or UNIX	14
Connection attributes	15
DSN-less connections	17
Logging	18
ODBC Driver Manager logging on Linux or UNIX	18
Easysoft ODBC-Sybase Driver logging on Linux and UNIX	18
Client applications.....	19
Oracle.....	20
Connecting Sybase to Oracle on Linux and UNIX.....	20
LibreOffice.....	23
Go.....	24
Node.js.....	25
Perl.....	27
PHP.....	30
Python.....	32
R.....	34
About the Easysoft ODBC-Sybase Driver	36
ODBC API and scalar functions.....	37
API functions	37
Scalar functions	39
Data type mapping	42
SQL support.....	43
Example queries.....	43
Example inserts, updates, and deletes	49
Index	52

Getting started

This section shows you how to install the Easysoft ODBC-Sybase Driver and configure the ODBC data source that stores the connection details for your Sybase database. You're then ready to work with Sybase data in your application.

- [Installing the Easysoft ODBC-Sybase Driver](#)
- [Connecting to Sybase](#)
- [Logging](#)

Installing the Easysoft ODBC-Sybase Driver

Install the Easysoft ODBC-Sybase Driver on the computer where the application you want to connect to Sybase is running.

Installing on Linux or UNIX

The installation can be done by anyone with root access.

1. [Download the Easysoft ODBC-Sybase Driver distribution for your client application platform.](#)

If your [client application is 64-bit](#), choose the 64-bit driver distribution from the **Platforms** list. If your [client application](#) is 32-bit, choose the 32-bit driver distribution from the **Platforms** list.

2. Copy the distribution to a temporary directory on the machine where the application you want to connect to Sybase is installed.
3. Unpack the distribution and cd into the resultant directory.
4. As root, run:

```
./install
```

5. Follow the onscreen instructions to progress through the installation.

Further information

- [Preinstallation requirements](#)
- [What you can install](#)
- [Where to install](#)
- [Changes made to your system](#)
- [Installing alongside other existing Easysoft product installations](#)
- [Gathering information required during the installation](#)
- [Unpacking the distribution](#)
- [License to use](#)
- [Answering questions during the installation](#)
- [Running the installer](#)
- [Locating or installing unixODBC](#)
- [Installing the Easysoft ODBC driver](#)
- [Licensing](#)
- [Post installation steps for non-root installations](#)

Preinstallation requirements

To install the Easysoft ODBC-Sybase Driver 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 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'll 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-Sybase Driver in /usr/local/easysoft.
- Install the Easysoft ODBC-Sybase 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-Sybase Driver elsewhere and set the EASYSOFT_ROOT environment variable. For more information about setting the EASYSOFT_ROOT environment variable, refer to [Post installation steps for non-root installations](#).
- An ODBC Driver Manager.

Easysoft ODBC-Sybase 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-Sybase 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.

We recommend that you install all components as the root user.

What you can install

This distribution contains:

- The Easysoft ODBC-Sybase Driver.
- The unixODBC Driver Manager.

You need an ODBC Driver Manager to use the Easysoft ODBC-Sybase Driver from your applications. The distribution therefore contains the unixODBC Driver Manager. Most (if not all) UNIX and Linux applications 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-Sybase 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 ODBC Driver Manager in the standard places. If you have installed it in a non-standard location, the installation script prompts 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're 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 tries 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, refer to [Post installation steps for non-root installations](#).

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

The 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-Sybase 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.)
2. The installation script installs an example data source into unixODBC. This data source will be added to your SYSTEM odbc.ini file. You can locate your SYSTEM odbc.ini file by using odbcinst -j.
3. Dynamic linker. On operating systems where the dynamic linker has a file listing locations for shared objects (Linux and FreeBSD), 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.
 - On FreeBSD this is usually the file /etc/defaults/rc.conf.

Installing alongside other existing Easysoft product installations

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 are 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 get 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're 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're 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.

Unpacking the distribution

The distribution for UNIX and Linux platforms is a tar file. To extract the installation files from the tar file, use:

```
tar -xvf odbc-sybase-1.3.0-linux-x86_64-unicode.tar
```

This creates 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 into the directory created by unpacking the tar file to run the installation script. For example:

```
# cd odbc-sybase-1.3.0-linux-x86_64-unicode
```

License to use

The end-user license agreement (EULA) is in the file license.txt. Be sure to understand the terms of the agreement before continuing, as you're required to accept the license terms at the start of the installation.

Answering questions during the installation

Throughout the installation, you're prompted to answer some questions. In each case, the default choice displays in square brackets and you need only press Enter to accept the default. If there are alternative responses, these are 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

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. We recommend installing as the root user. (If you're concerned about the changes that will be made to your system, refer to [Changes made to your system.](#))

To start the installation, run:

```
./install
```

You need to:

- Confirm your acceptance of the license agreement by typing "yes" or "no". For more information about the license agreement, refer to [License to use](#).
- Supply the location where the software is to be installed.

We recommend accepting the default installation path.

For more information, refer to [Where to install](#).

Locating or installing unixODBC

We strongly recommend you use the unixODBC Driver Manager because:

- The installation script is designed to work with unixODBC and can automatically add Easysoft ODBC-Sybase Driver and data sources during the installation.

- Most applications and interfaces that support ODBC are compatible with unixODBC. The Easysoft ODBC-Sybase Driver and any data sources that you add during the installation are automatically 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-Sybase Driver running under unixODBC. It also means that if you find a problem in unixODBC, it's 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 displays:

```
Found unixODBC under path and it is version n.n.n
```

2. If the installation script can't 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 we provide.

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

Option	Description
<code>--prefix=/etc</code>	This means the default SYSTEM <code>odbc.ini</code> file where SYSTEM data sources are located is <code>/etc/odbc.ini</code> .
<code>--enable-drivers=no</code>	This means other ODBC drivers that come with unixODBC are not installed.
<code>--enable-iconv=no</code>	This means unixODBC does not look for <code>libiconv</code> . Warnings about not finding an <code>iconv</code> library were confusing our customers.
<code>--enable-stats=no</code>	Turns off 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.
<code>--enable-readline=no</code>	This turns off readline support in <code>isql</code> . We did this because it ties <code>isql</code> to the version of <code>libreadline</code> 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 turning on readline support makes <code>isql</code> unusable on these systems.
<code>--prefix=/usr/local/easysoft/unixODBC</code>	This installs unixODBC into <code>/usr/local/easysoft/unixODBC</code> .

Installing the Easysoft ODBC driver

The Easysoft ODBC-Sybase Driver installation script:

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

If the Easysoft ODBC-Sybase Driver is already registered with unixODBC, a warning displays that lists the drivers unixODBC knows about. If you're installing the Easysoft ODBC-Sybase Driver into a different directory than it was installed before, you need to edit your `odbcinst.ini` file after the installation and correct the Driver and Setup paths. unixODBC's `odbcinst` doesn't update these paths if a driver is already registered.

- Creates an example Easysoft ODBC-Sybase Driver data source. If unixODBC is installed and you registered the Easysoft ODBC-Sybase Driver with unixODBC, the installation script adds example data source to your `odbc.ini` file.

Licensing

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

Licenses are stored in `installation_path/easysoft/license/licenses`.

Important 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-Sybase Driver license:

```
Would you like to request a Easysoft ODBC-Sybase 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 `y`, the installation runs the `licshell` script.

To obtain a license automatically, you need to be connected to the Internet and allow outgoing connections to `license.easysoft.com` on port 8884. If you're not connected to the Internet or don't allow outgoing connections on port 8884, the License Client can create a license request file that you can email to us.

When you start the License Client, the following menu displays:

```
[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're installing. The License Client then runs a program that generates a key that's 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-Sybase 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.
- A reference number (also referred to as an authorization code). When applying for a trial license, press Enter when prompted for a reference number. This field only applies to full (paid) licenses.

You're then asked to choose 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

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

- [3] Cancel this operation

If you choose to obtain the license automatically, the License Client tries to open a TCP/IP connection to `license.easysoft.com` on port 8884 and send the details you supplied along with your machine 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 send the the request to us. The License daemon returns the license key, prints 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 sent the license request to us, we'll return a license key. Add this to the end of the file `installation_path/easysoft/license/licenses`.

Post installation steps for non-root installations

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

1. If you attempt to install the Easysoft ODBC-Sybase Driver under the unixODBC Driver Manager and you do not have write permission to unixODBC's `odbcinst.ini` file, the driver can't 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 `drv_template` file to `odbcinst.ini`. (`drv_template` is in the directory where the Easysoft 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 and FreeBSD), you 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 didn't install the Easysoft ODBC-Sybase 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 creates `/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 doesn't 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 add these paths to a system file run whenever someone logs. For example, */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.
```

Uninstalling on Linux or UNIX

There is no automated way to remove the Easysoft ODBC-Sybase Driver in this release. However, removal is quite simple. To do this:

1. Change directory to *installation_path*/easysoft and delete the product directory.
installation_path is the Easysoft ODBC-Sybase 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-Sybase Driver entry needs to be removed from the odbcinst.ini file. To check whether the Easysoft ODBC-Sybase Driver is configured under unixODBC, use odbcinst -q -d. If the command output contains [SYBASE], uninstall the driver from unixODBC by using:

```
odbcinst -u -d -n SYBASE
```

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

1. If you created any Easysoft ODBC-Sybase 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 SYBASE.
2. Remove the install.info for the Easysoft ODBC-Sybase Driver from the /usr/local/easysoft directory.

Connecting to Sybase

Applications that support ODBC interface with an ODBC Driver Manager, which is included with the operating system, and also the Easysoft ODBC driver distribution on some platforms. One of the jobs that the ODBC Driver Manager does is to manage ODBC data sources. A data source specifies which ODBC driver to load, which data store to connect to, and how to connect to it.

Before setting up a data source, you must have [successfully installed the Easysoft ODBC-Sybase Driver](#).

Connecting from Linux or UNIX

Creating an ODBC data source

There are two ways to create a data source to your Sybase data:

- Create a SYSTEM data source, which is available to anyone who logs on to the computer where the Easysoft ODBC-Sybase Driver is installed.
 - Or –
- Create a USER data source, which is only available to the user who is currently logged on to the computer where the Easysoft ODBC-Sybase Driver is installed.

By default, the Easysoft ODBC-Sybase Driver installation creates a sample SYSTEM data source named demo-sybase. If you're using the unixODBC included in the Easysoft ODBC-Sybase 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 Sybase, USER data sources must be created and edited in `$HOME/.odbc.ini`.

Notes

- 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.

The Driver attribute identifies the ODBC driver in the `odbcinst.ini` file to use for a data source. When the Easysoft ODBC-Sybase Driver is installed into unixODBC, it places a SYBASE entry into the `odbcinst.ini` file. You should always have `Driver = SYBASE` in your Easysoft ODBC-Sybase Driver data sources therefore.

To configure a Easysoft ODBC-Sybase Driver data source, in your `odbc.ini` file, you need to specify:

- The database name (Database).
- The database user name (User).
- The database password (Password).
- The host name or IP address of the computer on which the Sybase server is running (Server_Host).
- The port on which the Sybase server is listening, which is 4100 by default. For Sybase ASE Express Edition, the default port is 5000. For SAP SQL Anywhere 17, the default port is 2638 (Server_Port).

For example:

```
[demo-sybase]
Driver          = SYBASE
Database        =
User            = myuser
Password        = mypassword
Server_Host     = localhost
Server_Port     = 4100
```

The Easysoft ODBC-Sybase Driver must be able to find the following shared objects:

- 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/.

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.

The isql query tool lets you test your Easysoft ODBC-Sybase Driver data sources. To test the Easysoft ODBC-Sybase Driver connection:

1. Change directory into /usr/local/easysoft/unixODBC/bin.
2. Enter `./isql -v data_source`, where *data_source* is the name of the target data source.
3. At the prompt, enter an SQL query. For example:

```
SQL> SELECT * FROM systypes;
```

–Or–

4. Enter help to return a list of tables:

```
SQL> help
```

Connection attributes

These optional attributes may also be set in odbc.ini.

Attribute	Description
Description	Some applications display this to help users identify a particular data source.
METADATA_ID	When turned on (set to 1), the default value of the SQL_ATTR_METADATA_ID connection attribute is set to SQL_TRUE. By default, METADATA_ID is off.
METADATA_DONT_CHANGE_CASE	When turned on (set to 1), the case of the parameter values passed to metadata calls do not change.

16 Connection attributes

Attribute	Description
TEXTSIZE	Sets the maximum size, in bytes, of text or image data returned from the server. The default size is 32000. If data is larger than this value, the data will be truncated, without any indication that it has been truncated.
QUOTED_IDENTIFIERS	This attribute switches the Sybase database quoted_identifier setting to on. This enables support for quoted identifiers and also changes the appropriate SQLGetInfo values returned.
CHARSET	Sets the character encoding for the Easysoft ODBC-Sybase Driver. For example, CHARSET=ISO8859-15.
LANGUAGE	Sets the language for the Easysoft ODBC-Sybase Driver. For example, LANGUAGE=uk_english.

DSN-less connections

Some applications allow you to make an ODBC connection without configuring a data source. To do this, you supply a connection string that contains the ODBC driver name and other driver-specific attribute-value pairs.

Here's an example Easysoft ODBC-Sybase Driver connection string:

```
Driver={SYBASE};SERVER_HOST=myserver;SERVER_PORT=2638;DB=pubs;UID=myuser;PWD=mypass  
word;
```

Logging

If you report an issue to us, we may ask you to turn on ODBC Driver Manager or Easysoft ODBC-Sybase Driver logging, to help us diagnose the cause of the issue.

To turn on logging, refer to the following sections.

Note If your application is a service (for example, Oracle or SQL Server), you may need to restart the service before enabling logging takes effect. To do this on Linux or UNIX, use `service`, `systemctl`, or a vendor-supplied script. To do this on Windows, use the Windows **Services** app.

ODBC Driver Manager logging on Linux or UNIX

For the unixODBC Driver Manager, add the following 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's running the application to log has write permission to `TraceFile` (and to the directory containing it), otherwise no logging information will be produced.

Easysoft ODBC-Sybase Driver logging on Linux and UNIX

Driver manager trace files show all the ODBC calls an application makes, including their arguments and return values. Easysoft ODBC-Sybase Driver logging is specific to the Easysoft driver and is of most use when making a support call.

To turn on Easysoft ODBC-Sybase Driver logging, edit your ODBC data source in `odbc.ini`. For example:

```
[demo-sybase]
.
.
.
LOG = /tmp/easysoft-odbc-driver.log
```

The value shown in the example specifies a log file named `/tmp/easysoft-odbc-driver.log`. Ensure that the user who's running the application to log has write permission to the log file (and to the directory containing it), otherwise no logging information will be produced.

Client applications

How to work with Sybase data in some example applications and programming languages:

- [Oracle](#)
- [LibreOffice](#)
- [Go](#)
- [Node.js](#)
- [Perl](#)
- [PHP](#)
- [Python](#)
- [R](#)

Oracle

1. [Install the Easysoft ODBC-Sybase Driver](#) on same computer as Oracle.
2. [Configure an ODBC data source](#).
3. Follow the instructions for your Oracle platform.

Connecting Sybase to Oracle on Linux and UNIX

1. Create a DG4ODBC init file on your Oracle machine. To do this, change to the \$ORACLE_HOME\hs\admin directory. Create a copy of the file initdg4odbc.ora. Name the new file initSybase.ora.

Note In these instructions, replace \$ORACLE_HOME with the location of your Oracle HOME directory. For example, /u01/app/oracle/product/21c/dbhome_1.

2. Ensure these parameters and values are present in your init file:

```
HS_FDS_CONNECT_INFO = "Sybase"
HS_FDS_SUPPORT_STATISTICS = FALSE
```

Replace Sybase with the name of your Easysoft ODBC-Sybase Driver data source.

The HS_FDS_SUPPORT_STATISTICS parameter controls whether Oracle gathers statistics from the Sybase database at the same time as trying to query an ODBC connection. Sybase ASE does not support this behaviour. If you don't set the HS_FDS_SUPPORT_STATISTICS parameter to False, you may get an error similar to:

```
General error: connection is busy with results of another hstmt
```

3. Comment out the line that enables DG4ODBC tracing. For example:

```
#HS_FDS_TRACE_LEVEL = <trace_level>
```

4. Add an entry to \$ORACLE_HOME/network/admin/listener.ora that creates a SID_NAME for DG4ODBC. For example:

```
SID_LIST_LISTENER =
(SID_LIST =
(SID_DESC=
(SID_NAME=Sybase)
(ORACLE_HOME=$ORACLE_HOME)
(PROGRAM=dg4odbc)
(ENVS=LD_LIBRARY_PATH = /usr/local/easysoft/unixODBC/lib:
/usr/local/easysoft/lib)
)
)
```

Replace oracle_home_directory with the value of \$ORACLE_HOME. For example, /u01/app/oracle/product/21c/dbhome_1.

5. Add a DG4ODBC entry to \$ORACLE_HOME/network/admin/tnsnames.ora that specifies the SID_NAME created in the previous step. For example:

```
Sybase =
(DESCRIPTION =
```

```
(ADDRESS = (PROTOCOL = TCP)(HOST = oracle_host)(PORT = 1521))
(CONNECT_DATA =
  (SID = Sybase)
)
(HS = OK)
)
```

Replace oracle_host with the host name of your Oracle machine.

6. Start (or restart) the Oracle Listener:

```
cd $ORACLE_HOME/bin
./lsnrctl stop
./lsnrctl start
```

7. Connect to your Oracle database in SQL*Plus.

8. In SQL*Plus, create a database link for Sybase. For example:

```
CREATE PUBLIC DATABASE LINK SybaseLink
  CONNECT TO "dbuser" IDENTIFIED BY "dbpassword"
  USING 'Sybase';
```

Replace dbuser and dbpassword with your backend user name and password, if applicable.

9. Try querying and updating your Sybase data. For example:

```
SELECT "Surname" FROM "Customers"@SybaseLink;

DECLARE
  num_rows integer;
BEGIN
  num_rows:=DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE@SybaseLink
  ('INSERT INTO Customers (Surname, GivenName, City, State, Country) VALUES
  ('Devlin', 'Michaels', 'Kingston', 'NJ', 'USA')));
END;
/

DECLARE
  num_rows integer;
BEGIN
  num_rows:=DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE@SybaseLink
  ('UPDATE "Customers" SET "Surname" = ''Jones'' WHERE "ID" = ''667''');
END;
/

DECLARE
  num_rows integer;
BEGIN
  num_rows:=DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE@SybaseLink
  ('DELETE from "Customers" WHERE ID = ''667''');
END;
/
```

Notes

- If you have problems connecting to Sybase from Oracle, enable DG4ODBC tracing and check the trace files written to the \$ORACLE_HOME/hs/trace directory. To enable DG4ODBC tracing, add the line HS_FDS_TRACE_LEVEL = DEBUG to initSybase.ora and then start or restart the Oracle listener. If the trace directory does not exist, create it.

LibreOffice

1. [Install the Easysoft ODBC-Sybase Driver](#) on same computer as LibreOffice.
2. [Configure an ODBC data source](#).
3. Choose **File > New > Database**.
4. Choose **Connect to an existing database**.
5. Choose **ODBC** in the list, and then choose **Next**.
6. Choose **Browse**, double-click your data source, and then choose **Next**.
7. If your database requires a database user name, enter it in the **User name** box. If this user needs to supply a password choose the **Password required** check box.
8. Choose **Finish**.
9. Save the database when prompted.

The database opens in a new Base window. From here you can access your data.

10. In the left pane of the database window, choose the **Tables** icon to display a hierarchy of tables. Enter the database password if prompted, and then choose **OK**.
11. To retrieve the data in a table, in the **Tables** pane, double-click a table.
12. Choose the **Queries** icon to create a query.

Use any of the methods listed in the **Tasks** pane to create a query.

Go

1. [Install the Easysoft ODBC-Sybase Driver](#) on same computer as Go.
2. [Configure an ODBC data source](#).
3. Install the `odbc` package for Go:

```
go mod init test
go get github.com/alexbrainman/odbc
```

4. Create and then use Go to run this script, which retrieves some Sybase data:

```
package main

import (
    _ "github.com/alexbrainman/odbc"
    "database/sql"
    "log"
)

func main() {
    // Replace the DSN value with the name of your ODBC data source.
    db, err := sql.Open("odbc",
        "DSN=Sybase")
    if err != nil {
        log.Fatal(err)
    }

    var (
        name string
    )

    rows, err := db.Query("SELECT Surname FROM Customers")
    if err != nil {
        log.Fatal(err)
    }
    defer rows.Close()
    for rows.Next() {
        err := rows.Scan(&name)
        if err != nil {
            log.Fatal(err)
        }
        log.Println(name)
    }
    err = rows.Err()
    if err != nil {
        log.Fatal(err)
    }

    defer db.Close()
}
```


Node.js

1. [Install the Easysoft ODBC-Sybase Driver](#) on same computer as Node.js.
2. [Configure an ODBC data source](#).
3. Install the `odbc` module for Node.js:

```
npm install odbc
```

4. Create and then use Node.js to run this script, which retrieves some Sybase data:

```
const odbc = require('odbc');
// Replace Sybase with the name of your Easysoft ODBC-Sybase Driver
// data source.
const connection = odbc.connect('DSN=Sybase', (error, connection) => {
  connection.query('SELECT Surname FROM Customers', (error, result) => {
    if (error) { console.error(error) }
    console.log(result);
  });
});
```

5. This script retrieves the tables and views in your Easysoft ODBC-Sybase Driver data source:

```
const odbc = require('odbc');
const connection = odbc.connect('DSN=Sybase', (error, connection) => {
  connection.tables(null, null, null, null, (error, result) => {
    if (error) { return; }
    const util = require('util');
    console.log(util.inspect(result, {maxLength: null, depth:null}))
  });
});
```

6. This script retrieves the names of the columns in these tables and views:

```
const odbc = require('odbc');
const connection = odbc.connect('DSN=Sybase', (error, connection) => {
  connection.columns(null, null, null, null, (error, result) => {
    if (error) { return; }
    const util = require('util');
    console.log(util.inspect(result, {maxLength: null, depth:null}))
  });
});
```

7. These scripts insert, update, and then delete some Sybase data:

```
const odbc = require("odbc");
const connection = odbc.connect("DSN=Sybase", (error, connection) => {
  connection.query("INSERT INTO
Customers (
  Surname,
  GivenName,
  City,
  State,
  Country
```

```
)
VALUES
(
  'Devlin',
  'Michaels',
  'Kingston',
  'NJ',
  'USA'
)", (error, result) => {
  if (error) { console.error(error) }
  console.log(result);
});

});

const odbc = require("odbc");
const connection = odbc.connect("DSN=Sybase", (error, connection) => {
  connection.query("UPDATE Customers SET Surname = 'Jones' WHERE ID = '667'",
(error, result) => {
  if (error) { console.error(error) }
  console.log(result);
});
});

const odbc = require("odbc");
const connection = odbc.connect("DSN=Sybase", (error, connection) => {
  connection.query("DELETE FROM Customers WHERE ID = '667'", (error, result) =>
{
  if (error) { console.error(error) }
  console.log(result);
});
});
});
```

Perl

1. [Install the Easysoft ODBC-Sybase Driver](#) on same computer as Perl.
2. [Configure an ODBC data source](#).
3. Check whether your Perl distribution supports ODBC:

```
perl -e 'use DBD::ODBC;'
```

4. Do one of the following:
 - If you get no output, your Perl distribution supports ODBC. Skip to the next step.
 - If you get:

```
Can't locate DBD/ODBC.pm
```

you need to [install DBD::ODBC](#) before you can use the Easysoft ODBC-Sybase Driver to connect to Sybase.

5. Create and then use Perl to run this script, which retrieves some Sybase data:

```
use strict;
use DBI;
# Replace Sybase with the name of your Easysoft ODBC-Sybase Driver data source.
my $dbh = DBI-> connect('dbi:ODBC:Sybase');

my $sql = "SELECT Surname FROM Customers";

my $sth = $dbh->prepare($sql)
    or die "Can't prepare statement: $DBI::errstr";

$sth->execute();

my($Col);

# Fetch and display the result set values.
while(($Col) = $sth->fetchrow()){
    print("$Col\n");
}

$dbh->disconnect if ($dbh);
```

6. This script retrieves the tables and views in your Easysoft ODBC-Sybase Driver data source:

```
use strict;
use DBI;
my $dbh = DBI-> connect('dbi:ODBC:Sybase');

my $sth = $dbh->table_info()
    or die "Can't prepare statement: $DBI::errstr";

my @row;

while (@row = $sth->fetchrow_array) {
    print join(", ", @row), "\n";
}
```

```
}
$dbh->disconnect if ($dbh);
```

7. This script retrieves the names of the columns in these tables and views:

```
use strict;
use DBI;
my $dbh = DBI-> connect('dbi:ODBC:Sybase');

my $sth = $dbh->column_info('', '', '', '')
    or die "Can't prepare statement: $DBI::errstr";

my @row;
while (@row = $sth->fetchrow_array) {
    print join(", ", @row), "\n";
}

$dbh->disconnect if ($dbh);
```

8. These scripts insert, update, and then delete some Sybase data:

```
use strict;
use DBI;
my $dbh = DBI-> connect('dbi:ODBC:Sybase');

my $sth = $dbh->prepare(q/INSERT INTO Customers (Surname, GivenName, City, State,
Country) VALUES (?, ?, ?, ?, ?)/)
    or die "Can't prepare statement: $DBI::errstr";

$sth->execute('Devlin', 'Michaels', 'Kingston', 'NJ', 'USA');

$dbh->disconnect if ($dbh);

use strict;
use DBI;
my $dbh = DBI-> connect('dbi:ODBC:Sybase');

my $sth = $dbh->prepare('UPDATE Customers SET Surname = \'Jones\' WHERE ID = ?')
    or die "Can't prepare statement: $DBI::errstr";

$sth->execute('667');

$dbh->disconnect if ($dbh);

use strict;
use DBI;
my $dbh = DBI-> connect('dbi:ODBC:Sybase');

my $sth = $dbh->prepare('DELETE FROM Customers WHERE ID = ?')
    or die "Can't prepare statement: $DBI::errstr";
```

```
$sth->execute('667');  
  
$dbh->disconnect if ($dbh);
```

Further information

- [Perl DBI DBD::ODBC tutorial: Drivers, data sources, and connection](#)

PHP

1. [Install the Easysoft ODBC-Sybase Driver](#) on same computer as PHP.
2. [Configure an ODBC data source](#).
3. Check whether your PHP distribution supports ODBC. In php.ini, make sure there is no comment character (;) before the extension_dir and extension=odbc settings (;extension_dir=directory becomes extension_dir=directory and ;extension=odbc becomes extension=odbc).
4. Create and then use PHP to run this script, which retrieves some Sybase data:

```
<?php
// Replace Sybase with the name of your Easysoft ODBC-Sybase Driver data
source.
// If your database requires a user name and password, supply them in the
odbc_connect_call.
$con = odbc_connect("Sybase", "", "");
$stmt = odbc_exec($con, "SELECT * FROM Customers");
// You may need to change the capitalisation of Surname to all upper case or
all lower case.
while ($row = odbc_fetch_array($stmt)) {
    echo "Surname = ", $row["Surname"], "\n";
}
odbc_close($con);
?>
```

5. This script retrieves the tables and views in your Easysoft ODBC-Sybase Driver data source:

```
<?php
$con = odbc_connect("Sybase", "", "");
$tables = odbc_tables($con);
while (($row = odbc_fetch_array($tables))) {
    print_r($row);
}
odbc_close($con);
?>
```

6. This script retrieves the names of the columns in these tables and views:

```
<?php
$con = odbc_connect("Sybase", "", "");
$columns = odbc_columns($con);
while (($row = odbc_fetch_array($columns))) {
    print_r($row);
}
odbc_close($con);
?>
```

7. These scripts insert, update, and then delete some Sybase data:

```
<?php
$conx = odbc_connect("Sybase", "", "");
$stmt = odbc_prepare($conx, "INSERT INTO Customers (Surname, GivenName, City,
State, Country) VALUES (?, ?, ?, ?, ?)");
```

```
$success = odbc_execute($stmt, array('Devlin', 'Michaels', 'Kingston', 'NJ',  
'USA'));  
odbc_close($cnx);  
?>  
  
<?php  
$cnx = odbc_connect("Sybase", "", "");  
$stmt = odbc_prepare($cnx, "UPDATE Customers SET Surname = 'Jones' WHERE ID =  
?");  
$success = odbc_execute($stmt, array('667'));  
odbc_close($cnx);  
?>  
  
<?php  
$cnx = odbc_connect("Sybase", "", "");  
$stmt = odbc_prepare($cnx, "DELETE FROM Customers WHERE ID = ?");  
$success = odbc_execute($stmt, array('667'));  
odbc_close($cnx);  
?>
```

Further information

- [Easysoft PHP tutorials and code samples](#)

Python

1. [Install the Easysoft ODBC-Sybase Driver](#) on same computer as Python.
2. [Configure an ODBC data source](#).
3. Check whether your Python distribution supports ODBC.

```
pip list
```

If you don't have pip installed:

```
curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
python get-pip.py
```

4. Do one of the following:
 - If the output contains pyodbc, your Python distribution supports ODBC. Skip to the next step.
 - If the output does not contain pyodbc, use pip to install this module:

```
pip install pyodbc
```

5. Create and then use Python to run this script, which retrieves some Sybase data:

```
import pyodbc

# Replace Sybase with the name of your Easysoft ODBC-Sybase Driver data source.
cnxn = pyodbc.connect("DSN=Sybase")
cursor = cnxn.cursor()
sql = "SELECT Surname FROM Customers"
cursor.execute(sql)
rows = cursor.fetchall()
# You may need to change the capitalisation of Surname to all upper case or all
lower case.
for row in rows:
    print(row.Surname)
exit()
```

6. This script retrieves the tables and views in your Easysoft ODBC-Sybase Driver data source:

```
import pyodbc

# Replace Sybase with the name of your Easysoft ODBC-Sybase Driver data source.
cnxn = pyodbc.connect("DSN=Sybase")
cursor = cnxn.cursor()
cursor.tables()
rows = cursor.fetchall()
for row in rows:
    print(row.table_name)
exit()
```

7. This script retrieves the names of the columns in these tables and views:

```
import pyodbc

# Replace Sybase with the name of your Easysoft ODBC-Sybase Driver data source.
```



```
cnxn = pyodbc.connect("DSN=Sybase")
cursor = cnxn.cursor()
cursor.columns()
rows = cursor.fetchall()
for row in rows:
    print(row.table_name, row.column_name)
exit()
```

8. These scripts insert, update, and then delete some Sybase data:

```
import pyodbc

cnxn = pyodbc.connect("DSN=Sybase")
cursor = cnxn.cursor()
sql = "INSERT INTO Customers (Surname, GivenName, City, State, Country) VALUES
(?, ?, ?, ?, ?)"
cursor.execute(sql, 'Devlin', 'Michaels', 'Kingston', 'NJ', 'USA')
cursor.commit()
exit()
```

```
import pyodbc

cnxn = pyodbc.connect("DSN=Sybase")
cursor = cnxn.cursor()
sql = "UPDATE Customers SET Surname = 'Jones' WHERE ID = ?"
cursor.execute(sql, '667')
cursor.commit()
exit()
```

```
import pyodbc

cnxn = pyodbc.connect("DSN=Sybase")
cursor = cnxn.cursor()
sql = "DELETE FROM Customers WHERE ID = ?"
cursor.execute(sql, '667')
cursor.commit()
exit()
```

Further information

- [Easysoft Python tutorials and code samples](#)

R

1. [Install the Easysoft ODBC-Sybase Driver](#) on same computer as R.
2. [Configure an ODBC data source](#).
3. In R Console, check whether your R distribution supports ODBC.

```
library("RODBC")
```

4. Do one of the following:
 - If you get no output, you have the ODBC library for R. Skip to the next step.
 - If you get an "there is no package" error, install the ODBC library for R:

```
install.packages("RODBC")
```

5. Create and then use R to run this script, which retrieves some Sybase data:

```
library("RODBC")
# Replace Sybase with the name of your Easysoft ODBC-Sybase Driver data source.
ch <- odbcConnect("Sybase")
sqlQuery(ch, paste("SELECT Surname FROM Customers"))
odbcClose(ch)
quit()
```

6. This script retrieves the tables and views in your Easysoft ODBC-Sybase Driver data source:

```
library("RODBC")
# Replace Sybase with the name of your Easysoft ODBC-Sybase Driver data source.
ch <- odbcConnect("Sybase")
sqlTables(ch)
odbcClose(ch)
quit()
```

7. This script retrieves the names of the columns in the specified table or view:

```
library("RODBC")
# Replace Sybase with the name of your Easysoft ODBC-Sybase Driver data source.
ch <- odbcConnect("Sybase")
# You may need to change the capitalisation of Customers to all upper case or all
lower case.
sqlColumns(ch, sqtable="Customers")
odbcClose(ch)
quit()
```

8. These scripts insert, update, and then delete some Sybase data:

```
library("RODBC")
ch <- odbcConnect("Sybase")
sqlQuery(ch, paste("INSERT INTO Customers (Surname, GivenName, City, State,
Country) VALUES ('Devlin', 'Michaels', 'Kingston', 'NJ', 'USA')"))
odbcClose(ch)
quit()

library("RODBC")
```

```
ch <- odbcConnect("Sybase")
sqlQuery(ch, paste("UPDATE Customers SET Surname = 'Jones' WHERE ID = '667'"))
odbcClose(ch)
quit()

library("RODBC")
ch <- odbcConnect("Sybase")
sqlQuery(ch, paste("DELETE FROM Customers WHERE ID = '667'"))
odbcClose(ch)
quit()
```

About the Easysoft ODBC-Sybase Driver

The Easysoft ODBC-Sybase Driver provides real-time access to Sybase data from any application that supports ODBC.

- [ODBC API and scalar functions](#)
- [Data type mapping](#)
- [SQL support](#)

ODBC API and scalar functions

API functions

Use this table to find out what ODBC API functions the Easysoft ODBC-Sybase Driver supports:

Function	Status
SQLAllocConnect	Supported
SQLAllocEnv	Supported
SQLAllocHandle	Supported
SQLAllocStmt	Supported
SQLBindCol	Supported
SQLBindParameter	Supported
SQLBrowseConnect	Not supported
SQLBulkOperations	Not supported
SQLCancel	Supported
SQLCloseCursor	Supported
SQLColAttribute	Supported
SQLColAttributes	Supported
SQLColumnPrivileges	Supported
SQLColumns	Supported
SQLConnect	Supported
SQLCopyDesc	Not supported
SQLDataSources	Supported
SQLDescribeCol	Supported
SQLDescribeParam	Not supported
SQLDisconnect	Supported
SQLDriverConnect	Supported
SQLDrivers	Supported
SQLEndTran	Supported
SQLError	Supported
SQLExecDirect	Supported
SQLExecute	Supported
SQLExtendedFetch	Supported
SQLFetch	Supported
SQLFetchScroll	Supported
SQLForeignKeys	Supported
SQLFreeConnect	Supported

Function	Status
SQLFreeEnv	Supported
SQLFreeHandle	Supported
SQLFreeStmt	Supported
SQLGetConnectAtt	Supported
SQLGetConnectOption	Supported
SQLGetCursorName	Supported
SQLGetData	Supported
SQLGetDescField	Supported
SQLGetDescRec	Supported
SQLGetDiagField	Supported
SQLGetDiagRec	Supported
SQLGetEnvAttr	Supported
SQLGetFunctions	Supported
SQLGetInfo	Supported
SQLGetStmtAttr	Supported
SQLGetStmtOption	Supported
SQLGetTypeInfo	Supported
SQLMoreResults	Supported
SQLNativeSql	Supported
SQLNumParams	Supported
SQLNumResultCols	Supported
SQLParamData	Supported
SQLParamOptions	Supported
SQLPrepare	Supported
SQLPrimaryKeys	Supported
SQLProcedureColumns	Supported
SQLProcedures	Supported
SQLPutData	Supported
SQLRowCount	Supported
SQLSetConnectAttr	Supported
SQLSetConnectOption	Supported
SQLSetCursorName	Supported
SQLSetDescField	Supported
SQLSetDescRec	Supported
SQLSetEnvAttr	Supported

Function	Status
SQLSetParam	Supported
SQLSetPos	Supported
SQLSetScrollOptions	Supported
SQLSetStmtOption	Supported
SQLSetStmtOption	Supported
SQLSetStmtAttr	Supported
SQLStatistics	Supported
SQLTablePrivileges	Supported
SQLTables	Supported
SQLTransact	Supported

Scalar functions

The Easysoft ODBC-Sybase Driver supports a number of scalar functions:

- [String functions](#)
- [Numeric functions](#)
- [Time, date, and interval functions](#)
- [System functions](#)
- [Conversion functions](#)

Use either the SQL-92 or ODBC syntax with scalar functions. For example:

```

SELECT
    Invoice_Id,
    Customer_Name,
    EXTRACT(YEAR FROM Due_Date) as "Year"
FROM
    Invoice

SELECT
    Invoice_Id,
    Customer_Name,
    {fn EXTRACT(YEAR FROM Due_Date)} as "Year"
FROM
    Invoice

```

String functions

The Easysoft ODBC-Sybase Driver supports these [string](#) functions:

- [ASCII\(*string_exp*\)](#)
- [CHAR\(*code*\)](#)
- [CHAR_LENGTH\(*string_exp*\)](#)
- [DIFFERENCE\(*string_exp1*, *string_exp2*\)](#)
- [LTRIM\(*string_exp*\)](#)
- [RIGHT\(*string_exp*, *count*\)](#)
- [RTRIM\(*string_exp*\)](#)

- `SOUNDEX(string_exp)`
- `SPACE(count)`
- `SUBSTRING(string_exp, start, length)`

Numeric functions

The Easysoft ODBC-Sybase Driver supports these [numeric](#) functions:

- `ABS(numeric_exp)`
- `ACOS(float_exp)`
- `ASIN(float_exp)`
- `ATAN(float_exp)`
- `CEILING(numeric_exp)`
- `COS(float_exp)`
- `COT(float_exp)`
- `DEGREES(numeric_exp)`
- `EXP(float_exp)`
- `FLOOR(numeric_exp)`
- `LOG(float_exp)`
- `LOG10(float_exp)`
- `PI()`
- `POWER(numeric_exp, integer_exp)`
- `RADIANS(numeric_exp)`
- `RAND([integer_exp])`
- `ROUND(numeric_exp, integer_exp)`
- `SIGN(numeric_exp)`
- `SIN(float_exp)`
- `SQRT(float_exp)`
- `TAN(float_exp)`
- `TRUNCATE(numeric_exp, integer_exp)`

Time, date, and interval functions

The Easysoft ODBC-Sybase Driver supports these [time, date, and interval](#) functions:

- `CURRENT_DATE()`
- `CURRENT_TIME([time-precision])`
- `CURRENT_TIMESTAMP([timestamp-precision])`
- `CURDATE()`
- `CURTIME()`
- `DAYNAME(date_exp)`
- `EXTRACT(extract-field FROM extract-source)`
- `HOUR(time_exp)`
- `MINUTE(time_exp)`
- `MONTH(date_exp)`
- `MONTHNAME(date_exp)`
- `QUARTER(date_exp)`
- `SECOND(time_exp)`
- `WEEK(date_exp)`
- `YEAR(date_exp)`

System functions

The Easysoft ODBC-Sybase Driver supports these [system](#) functions:

- IFNULL(*exp*, *value*)
- USER()

Conversion functions

The Easysoft ODBC-Sybase Driver supports both the [SQL-92 CAST](#) function and the [ODBC CONVERT](#) function for conversion between compatible data types.

Data type mapping

The Easysoft ODBC-Sybase Driver maps Sybase data types to ODBC data types in this way:

Sybase data type	ODBC data type
BIGINT	SQL_BIGINT
BINARY	SQL_BINARY
BIT	SQL_BIT
CHAR	SQL_CHAR
DATE	SQL_DATE
DATETIME	SQL_DATE SQL_TIMESTAMP SQL_TIME
DECIMAL	SQL_NUMERIC
FLOAT	SQL_DOUBLE
IMAGE	SQL_LONGVARBINARY
INT	SQL_INTEGER
MONEY	SQL_NUMERIC
NUMERIC	SQL_NUMERIC
REAL	SQL_REAL
SMALLDATETIME	SQL_TIMESTAMP
SMALLINT	SQL_SMALLINT
SMALLMONEY	SQL_NUMERIC
SYSNAME	SQL_VARCHAR
TEXT	SQL_LONGVARCHAR
TIME	SQL_TIME
TIMESTAMP	SQL_TIMESTAMP
TINYINT	SQL_TINYINT
UNSIGNED BIGINT	SQL_GUID
UNSIGNED INT	SQL_WVARCHAR
UNSIGNED SMALLINT	SQL_WLONGVARCHAR
VARBINARY	SQL_BINARY
VARCHAR	SQL_VARCHAR

SQL support

The Easysoft ODBC-Sybase Driver supports these SQL statements, clauses, and operators:

- SELECT
- SELECT DISTINCT
- WHERE
- ORDER BY
- AND
- OR
- NOT
- INSERT INTO
- NULL
- UPDATE
- DELETE
- TOP
- MIN
- MAX
- COUNT
- SUM
- AVG
- LIKE
- WILDCARDS
- IN
- BETWEEN
- ALIASES
- JOINS
- UNION
- GROUP BY
- HAVING
- EXISTS
- CASE

Example queries

- To fetch all records from a table, use the asterisk symbol (*) in your queries. For example:

```
SELECT * FROM Customers
```

- To only fetch records whose values are different, use DISTINCT. For example:

```
-- Which different sales regions are there?  
SELECT DISTINCT Region AS Different_Regions FROM SalesOrders  
-- How many different sales regions are there?  
SELECT COUNT(DISTINCT Region) AS Different_Regions FROM SalesOrders
```

44 Example queries

- To filter records, use WHERE. For example:

```
SELECT
    OrderDate,
    SalesRepresentative
FROM
    SalesOrders
WHERE
    Region = 'Eastern'

SELECT
    OrderDate,
    SalesRepresentative
FROM
    SalesOrders
WHERE
    Region = 'Eastern'
    OR Region = 'Western'

SELECT
    OrderDate,
    SalesRepresentative
FROM
    SalesOrders
WHERE
    Region = 'Eastern'
    AND {fn YEAR(OrderDate)} = 2024
```

You can also supply a WHERE clause value as a parameter. For example, to do this in [Python](#):

```
cursor.execute("SELECT
    OrderDate,
    SalesRepresentative
FROM
    SalesOrders
WHERE
    Region = ?", ['Eastern'])
```

- To fetch records that don't match the WHERE clause pattern use NOT. For example:

```
SELECT
    OrderDate,
    SalesRepresentative
FROM
    SalesOrders
WHERE
    NOT Region = 'Eastern'
```

- To sort the result set in either ascending or descending order, use ORDER BY. For example:

```
SELECT
    *
FROM
    SalesOrders
ORDER BY
    OrderDate ASC

SELECT
    *
FROM
    Contacts
ORDER BY
    (
        CASE
            WHEN Surname IS NULL THEN Title
            ELSE Surname
        END
    );
```

46 Example queries

- To group a result set into summary rows, use GROUP BY. For example:

```
SELECT
    COUNT(Id) As "Number",
    ProductID
FROM
    SalesOrderItems
GROUP BY
    ProductID

SELECT
    COUNT(Id) As "Number",
    ProductID
FROM
    SalesOrderItems
GROUP BY
    ProductID
HAVING
    COUNT(Id) > 100;
```

- To do calculations based on result set vales, use the SQL aggregate functions MIN(), MAX(), COUNT(), SUM(), and AVG(). For example:

```
SELECT Max(Quantity) FROM SalesOrderItems
SELECT Sum(Quantity) FROM SalesOrderItems
```

In addition, you can use these [scalar funtions](#).

- To convert between compatible data types, use CAST or CONVERT. For example:

```
SELECT CAST(Quantity AS Char(100))FROM SalesOrderItems
SELECT {fn CONVERT(Quantity, SQL_CHAR)} FROM SalesOrderItems
```

- To fetch records that contain column values between a given range, use BETWEEN. For example:

```
SELECT ProductID FROM SalesOrderItems WHERE Quantity BETWEEN 10 AND 20
```

- To combine the result set of two or more SELECT statements, use UNION. For example:

```
SELECT City FROM Contacts
UNION
SELECT City FROM Customers
ORDER BY City;
```

- To combine rows from two or more tables, use JOIN. For example:

```
SELECT SalesOrders.ID, Customers.Surname, SalesOrders.OrderDate
FROM SalesOrders
INNER JOIN Customers ON SalesOrders.CustomerID=Customers.ID;
```

- To fetch records that contain column values matching a search pattern, use LIKE. For example:

```
SELECT Surname, GivenName FROM Customers WHERE CompanyName LIKE 'R%'
SELECT Surname, GivenName FROM Customers WHERE CompanyName LIKE '_he'
```

- To search for columns without a value (NULL) or with a value (non NULL), use either IS NULL or IS NOT NULL. For example:

```
SELECT * FROM Customers WHERE CompanyName IS NULL
```

48 Example queries

- To specify multiple values in a WHERE clause, you can use IN as an alternative to OR. For example:

```
SELECT
    OrderDate,
    SalesRepresentative
FROM
    SalesOrders
WHERE
    Region = 'Eastern'
    OR Region = 'Western'
    OR Region = 'Central'
```

can be replaced with:

```
SELECT
    OrderDate,
    SalesRepresentative
FROM
    SalesOrders
WHERE
    Region IN ('Eastern', 'Western', 'Central')
```

- To set the maximum number of records to return, use TOP. For example:

```
SELECT TOP 10 * FROM Customers
```

- To test for the existence of records in a subquery, use EXISTS. For example:

```
SELECT
    Name
FROM
    Products
WHERE
    EXISTS (
        SELECT
            *
        FROM
            SalesOrderItems
        WHERE
            Products.ID = SalesOrderItems.ProductID
            AND Quantity < 20
    )
```


Example inserts, updates, and deletes

- To insert a Sybase record, use INSERT INTO. For example:

```
INSERT INTO
    Customers (
        Surname,
        GivenName,
        City,
        State,
        Country
    )
VALUES
    (
        'Devlin',
        'Michaels',
        'Kingston',
        'NJ',
        'USA'
    )
```

- Here's an Oracle linked table example:

```
DECLARE
    num_rows integer;
BEGIN
    num_rows:=DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE@SybaseLink
    ('INSERT INTO Customers (Surname, GivenName, City, State, Country) VALUES
    ('Devlin', 'Michaels', 'Kingston', 'NJ', 'USA')');
END;
/
```

- The Easysoft ODBC-Sybase Driver also supports parameterized inserts. Here's an example of doing this in [Perl](#):

```
my $sth = $dbh->prepare(q/INSERT INTO Customers (Surname, GivenName, City, State,
Country) VALUES (?, ?, ?, ?, ?)/)
    or die "Can't prepare statement: $DBI::errstr";

$sth->execute('Devlin', 'Michaels', 'Kingston', 'NJ', 'USA');
```

- To update a Sybase record, use UPDATE. For example:

```
UPDATE Customers
SET
    Surname = 'Jones'
WHERE
    Account_Id = '667'
```

The Easysoft ODBC-Sybase Driver also supports parameterized updates. Here's an example of doing this in [Perl](#):

```
my $sth = $dbh->prepare('UPDATE Customers SET Surname = \'Jones\' WHERE ID = ?')
    or die "Can't prepare statement: $DBI::errstr";

$sth->execute('667');
```

- To delete a Sybase record, use DELETE. For example:

```
-- Delete (mark inactive) a bank account  
DELETE FROM Customers WHERE ID = '667'
```

The Easysoft ODBC-Sybase Driver also supports parameterized deletes. Here's an example of doing this in [Python](#):

```
sql = "DELETE FROM Customers WHERE ID = ?"  
cursor.execute(sql, '667')
```

Index

A

attribute, [15](#)

B

Base

working with Sybase data, [23](#)

C

CHARSET attribute, [16](#)

connecting to Sybase, [14](#)

connection string attributes, [17](#)

D

data source attributes, [15](#)

data type mapping, [42](#)

Description attribute, [15](#)

DG4ODBC

working with Sybase data, [20](#)

DSN-less connections, [17](#)

E

Easysoft ODBC-Sybase Driver

adding ODBC data sources

on Linux or UNIX, [14](#)

connecting to Sybase, [14](#)

data source attributes, [15](#)

data type mapping, [42](#)

DSN-less connections, [17](#)

installing

on Linux or UNIX, [5](#)

logging, [18](#)

ODBC API support, [37](#)

scalar function support, [39](#)

SQL support, [43](#)

uninstalling

on Linux or UNIX, [13](#)

G

Go

working with Sybase data, [24](#)

I

installing the Easysoft ODBC-Sybase Driver, [5](#)

L

LANGUAGE attribute, [16](#)

LibreOffice

working with Sybase data, [23](#)

log files, [18](#)

M

METADATA_ID attribute, [15](#)

N

Node.js

working with Sybase data, [25](#)

O

ODBC API function support, [37](#)

ODBC connection string attributes, [17](#)

ODBC data sources

adding

on Linux or UNIX, [14](#)

Oracle

working with Sybase data, [20](#)

P

Perl

working with Sybase data, [27](#)

PHP

working with Sybase data, [30](#)

Python

working with Sybase data, [32](#)

Q

QUOTED_IDENTIFIERS attribute, [16](#)

R

R

working with Sybase data, [34](#)

S

scalar function support, [39](#)

SQL support, [43](#)

T

TEXTSIZE attribute, [16](#)

trace files, [18](#)

U

uninstalling

on Linux or UNIX, [13](#)