

# **Easysoft ODBC-Oracle Driver User's Guide**

This manual documents version 3.10.n of the Easysoft ODBC-Oracle 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 . . . . .	5
Installing the Oracle client . . . . .	6
Installing the Easysoft ODBC-Oracle Driver . . . . .	7
Installing on Linux or UNIX . . . . .	7
Uninstalling on Linux or UNIX . . . . .	15
Installing on Windows . . . . .	16
Uninstalling on Windows . . . . .	18
Connecting to Oracle . . . . .	19
Connecting from Linux or UNIX . . . . .	19
Connecting from Windows . . . . .	21
Connection attributes . . . . .	22
DSN-less connections . . . . .	30
Logging . . . . .	31
ODBC Driver Manager logging on Linux or UNIX . . . . .	31
Easysoft ODBC-Oracle Driver logging on Linux and UNIX . . . . .	31
ODBC Driver Manager logging on Windows . . . . .	31
Finding out what product version you have on Windows . . . . .	34
Client applications . . . . .	35
Microsoft Access . . . . .	36
Linking a table . . . . .	36
Importing a table . . . . .	36
Microsoft Excel . . . . .	37
Data Connection Wizard . . . . .	37
Microsoft Query . . . . .	37
PowerPivot . . . . .	37
Microsoft Power BI . . . . .	38
Microsoft SQL Server . . . . .	39
LibreOffice . . . . .	41
Go . . . . .	42
Node.js . . . . .	43
Perl . . . . .	45
PHP . . . . .	48
Python . . . . .	50
R . . . . .	52
About the Easysoft ODBC-Oracle Driver . . . . .	54
ODBC API and scalar functions . . . . .	55
API functions . . . . .	55
Scalar functions . . . . .	55
Cursor support . . . . .	56
Advanced Security . . . . .	57
Oracle Real Application Clusters (RAC) . . . . .	58
Transparent Application Failover (TAF) . . . . .	59
Database Resident Connection Pooling (DRCP) . . . . .	61
Background . . . . .	61
Network Protocols . . . . .	62
Materialized Views . . . . .	63
XA support . . . . .	64
Supported data types . . . . .	65
Finding out more about data types on Windows . . . . .	65
Example SQL statements . . . . .	66

---

Example queries .....	66
Example inserts, updates, and deletes .....	72
Index .....	75

## Getting started

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

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

## Installing the Oracle client

The Easysoft ODBC-Oracle Driver uses the Oracle client software to access Oracle. Either the Instant Client or standard Oracle client must be installed on the same computer as the Easysoft ODBC-Oracle Driver.

Although the Easysoft ODBC-Oracle Driver is compatible with both Oracle clients, we recommend that you use our driver with the Instant Client.

If your application is 64-bit, download the 64-bit version of the client. If your application is 32-bit, download the 32-bit version of the client.

Use SQL\*Plus, included in the client distribution to check that you can connect to your database. For example:

```
sqlplus system/p455w0rd@//localhost:1521/XE
```

**Important**

If you're unable to connect to Oracle with SQL\*Plus, contact your Oracle Database Administrator. If you cannot access your Oracle database with SQL\*Plus, you won't be able to access the database with the Easysoft ODBC-Oracle Driver.

At the prompt, enter a SELECT statement to test that you can retrieve some data:

```
SELECT * FROM dual;
```

To exit SQL\*Plus, enter exit.

# Installing the Easysoft ODBC-Oracle Driver

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

- [Installing on Linux or UNIX](#)
- [Uninstalling on Linux or UNIX](#)
- [Installing on Windows](#)
- [Uninstalling on Windows](#)

## Installing on Linux or UNIX

The installation can be done by anyone with root access.

1. [Download the Easysoft ODBC-Oracle 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 Oracle 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](#)
- [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-Oracle 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-Oracle Driver in /usr/local/easysoft.
  - Install the Easysoft ODBC-Oracle 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-Oracle 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-Oracle 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-Oracle 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-Oracle Driver.
- The unixODBC Driver Manager.

You need an ODBC Driver Manager to use the Easysoft ODBC-Oracle 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-Oracle 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-Oracle 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-oracle-3.10.0-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-oracle-3.10.0-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-Oracle Driver and data sources during the installation.
- Most applications and interfaces that support ODBC are compatible with unixODBC. The Easysoft ODBC-Oracle 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-Oracle 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-Oracle 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 is /etc/odbc.ini.
--enable-drivers=no	This means other ODBC drivers that come with unixODBC are not installed.
--enable-iconv=no	This means unixODBC does not look for libiconv. Warnings about not finding an iconv library were confusing our customers.
--enable-stats=no	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.
--enable-readline=no	This turns off readline support in isql. We did 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 turning on readline support makes isql unusable on these systems.

Option	Description
<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-Oracle Driver installation script:

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

If the Easysoft ODBC-Oracle Driver is already registered with unixODBC, a warning displays that lists the drivers unixODBC knows about. If you're installing the Easysoft ODBC-Oracle 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-Oracle Driver data source. If unixODBC is installed and you registered the Easysoft ODBC-Oracle 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-Oracle Driver license:

```
Would you like to request a Easysoft ODBC-Oracle 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-Oracle 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-Oracle 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-Oracle 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-Oracle 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-Oracle 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-Oracle 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-Oracle Driver entry needs to be removed from the odbcinst.ini file. To check whether the Easysoft ODBC-Oracle Driver is configured under unixODBC, use odbcinst -q -d. If the command output contains [ORACLE], uninstall the driver from unixODBC by using:

```
odbcinst -u -d -n ORACLE
```

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-Oracle 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 ORACLE.
2. Remove the install.info for the Easysoft ODBC-Oracle Driver from the /usr/local/easysoft directory.

### Installing on Windows

The Windows installation can be done by anyone with local administrator privileges.

1. [Download the Easysoft ODBC-Oracle Driver installer.](#)
2. Follow the onscreen instructions to progress through the installation wizard.

### Updating files that are in use

To avoid rebooting your computer, the Easysoft ODBC-Oracle Driver installer prompts you when files that it needs to update are in use by another application or service. This frees the locked files and allows the installation to complete without a system restart. The installer uses the **Restart Manager** to locate the applications that are using files that need updating. These applications are displayed in the **Files in Use** dialog box. To avoid a system restart, choose **Automatically close applications and attempt to restart them after setup is complete**. The Easysoft ODBC-Oracle Driver installer then uses **Restart Manager** to try to stop and restart each application or service in the list. If possible, **Restart Manager** restores applications to the same state that they were in before it shut them down.

### Licensing

By default, the installer starts the Easysoft License Manager, because you can't use the Easysoft ODBC-Oracle Driver until you have a license. If you choose not to run Easysoft License Manager as part of the installation process, run License Manager from the **Easysoft** group in the Windows **Start** menu when you're ready to license the Easysoft ODBC-Oracle Driver. These types of license are available:

- A free time-limited trial license, which gives you free and unrestricted use of the product for a limited period (usually 14 days).
- A full license if you have purchased the product. On purchasing the product you are given an authorization code, which you use to obtain a license.

To license the Easysoft ODBC-Oracle Driver:



1. In License Manager, enter your contact details.

You **must** complete the **Name**, **E-Mail Address**, and **Company** fields.

The e-mail address **must** be the same as the one used to register at the Easysoft web site. Otherwise, you won't be able to obtain a trial license.

2. Choose **Request License**.

You're prompted to choose a license type.

3. Do one of the following:

- For a trial license, choose **Time Limited Trial**, and then choose **Next**.

-Or-

- For a purchased license, choose **Non-expiring License**, and then choose **Next**.

4. Choose your product from the drop-down list when prompted, and then choose **Next**.

5. For a purchased license, enter your authorization code when prompted, and then choose **Next**.

6. Choose how to get your license when prompted.

7. Do one of the following:

- Choose **On-line Request** if your machine is connected to the internet and can make outgoing connections to port 8884.

With this method, License Manager automatically requests and then applies your license.

-Or-

- Choose **View Request**. Then open a web browser and go to [https://www.easysoft.com/support/licensing/trial\\_license.html](https://www.easysoft.com/support/licensing/trial_license.html) or [https://www.easysoft.com/support/licensing/full\\_license.html](https://www.easysoft.com/support/licensing/full_license.html), as appropriate. In the web page, enter your machine number (labelled **Number** in the license request). For purchased licenses, you also need to enter your authorization code (labelled **Ref** in the license request).

We'll automatically email your license to the email address you supplied in License Manager.

-Or-

- Choose **Email Request** to email your license request to our licensing team.

Once we've processed your request, we'll email your license to the email address you supplied in License Manager.

8. Close the License Manager windows and then choose **Finish**.

If you chose either **View Request** or **Email Request**, apply your license by double-clicking the email attachment when you get the license email from us. Alternatively, start License Manager from the **Easysoft** folder in the Windows **Start** menu. Then choose **Enter License** and paste the license in the space provided.

Once you've licensed the Easysoft ODBC-Oracle Driver, the installation is complete.

## Repairing the installation

The installer can repair a broken Easysoft ODBC-Oracle Driver installation. For example, you can use the installer to restore missing Easysoft ODBC-Oracle Driver files or registry keys. To do this:

1. In the Windows **taskbar**, enter Add or remove programs in the Windows **search** box.
2. Select Easysoft ODBC-Oracle Driver in the list, and then choose **Repair**.

## Uninstalling on Windows

This section explains how to remove the Easysoft ODBC-Oracle Driver from your system.

### Removing Easysoft ODBC-Oracle Driver data sources

Easysoft ODBC-Oracle Driver data sources are not removed when you uninstall the Easysoft ODBC-Oracle Driver. You don't therefore need to recreate your Easysoft ODBC-Oracle Driver data sources if you reinstall or upgrade. If you don't want to keep your Easysoft ODBC-Oracle Driver data sources, use Microsoft **ODBC Data Source Administrator** to remove them, **before** uninstalling the Easysoft ODBC-Oracle Driver:

1. In the Windows **taskbar**, enter Run in the Windows **search** box.
2. In the Windows **Run** dialog box, enter:

```
odbcad32.exe
```

3. Locate your data source in either the **User** or **System** tab.
4. Select the data source from the list, and then choose **Remove**.

If the **Remove** button isn't available, close **ODBC Data Source Administrator**, and then, in the Windows **Run** dialog box, enter:

```
%windir%\syswow64\odbcad32.exe
```

Repeat the previous two steps.

### Removing the Easysoft ODBC-Oracle Driver

1. In the Windows **taskbar**, enter Add or remove programs in the Windows **search** box.
2. Select Easysoft ODBC-Oracle Driver in the list, and then choose **Uninstall**.

**Note**

Easysoft product licenses are stored in the Windows registry. When you uninstall, your licenses are not removed, so you do not need to relicense the product if you reinstall or upgrade.

## Connecting to Oracle

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-Oracle Driver](#).

- [Connecting from Linux or UNIX](#)
- [Connecting from Windows](#)

## Connecting from Linux or UNIX

### Setting the environment

If you're using the Easysoft ODBC-Oracle Driver with the Instant Client, set LD\_LIBRARY\_PATH to point to the Instant Client directory. For example:

```
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/lib/instantclient_23
export LD_LIBRARY_PATH
```

If you're using the Easysoft ODBC-Oracle Driver with the standard Oracle client, set ORACLE\_HOME to point to the client directory. For example:

```
ORACLE_HOME=/home/oracle/OraHome1
export ORACLE_HOME
```

If you're using the standard Oracle client, set LD\_LIBRARY\_PATH to point to \$ORACLE\_HOME/lib. For example:

```
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/oracle/OraHome1/lib
export LD_LIBRARY_PATH
```

**Note** On some platforms, you need to use SHLIB\_PATH or LIBPATH rather than LD\_LIBRARY\_PATH.

## Creating an ODBC data source

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

- Create a SYSTEM data source, which is available to anyone who logs on to the computer where the Easysoft ODBC-Oracle 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-Oracle Driver is installed.

By default, the Easysoft ODBC-Oracle Driver installation creates a sample SYSTEM data source named demo-oracle. If you're using the unixODBC included in the Easysoft ODBC-Oracle 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 Oracle, 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-Oracle Driver is installed into unixODBC, it places a ORACLE entry into the odbcinst.ini file. You should always have Driver = ORACLE in your Easysoft ODBC-Oracle Driver data sources therefore.

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

- The Oracle database user name (User).
- The Oracle database password (Password).
- The SQL connect URL string (Database).

-Or-

If you're using the standard Oracle client:

- The tnsnames.ora service name for the database (Database).

Here's an Instant Client example:

```
[demo-oracle]
Driver       = ORACLE
Database     = //testhost:1521/testdb
User         = system
Password     = p455w0rd
```

Here's a standard client example:

```
[demo-oracle]
Driver       = ORACLE
Database     = testdb
User         = system
Password     = p455w0rd
```

The Easysoft ODBC-Oracle 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-Oracle Driver data sources. To test the Easysoft ODBC-Oracle 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 dual;
```

–Or–

4. Enter help to return a list of tables:

```
SQL> help
```

### Note

Some Easysoft ODBC-Oracle Driver distributions contain a diagnostic tool named checksys, which detects any configuration or environment problems and suggests corrective action. It's located in the `installation_dir/easysoft/oracle` directory. For example:

```
cd /usr/local/easysoft/oracle
./checksys -d data_source
```

## Connecting from Windows

### Creating an ODBC data source

1. In the Windows **taskbar search** box, enter "Run".
2. Do one of the following:
  - If your application is 64-bit, in the **Run** dialog box, enter:

```
odbcad32.exe
```

–Or–

- If your application is 32-bit, in the **Run** dialog box, enter:

```
%windir%\syswow64\odbcad32.exe
```

### Note

If you're not sure whether your application is 32-bit or 64-bit, start your application, then in Windows **Task Manager** check whether your application's process name contains (32-bit). For example, the process name for the 32-bit version of Excel is Microsoft Excel (32-bit); the process name for the 64-bit version of Excel is Microsoft Excel. On older versions of Windows, 32-bit applications contain \*32 in the process name rather than (32-bit).

For applications such as Oracle or SQL Server that run as a service, check the \*Background processes\* list rather than the **Apps** list in **Task Manager**.

If you're running a programming language from within a Windows command-line shell (for example, Command or PowerShell), in your shell, run the .exe file for the programming language. For example, run perl, php, python, or node. In **Task Manager**, expand the process list for **Windows Command Processor** or

**Windows PowerShell**, as appropriate, and check whether the process for your programming language contains (32-bit).

3. Do one of the following:
  - To create a data source that only the user you're currently logged in as can access, choose the **User** tab.  
If your application is a Windows service (for example, SQL Server or Oracle) creating a user data source won't work, unless the service is running as the same user you're logged in as.
  - To create a data source that all users on this computer can access, choose the **System** tab.
4. Choose **Add**.
5. In the list of ODBC drivers, select Easysoft ODBC-Oracle Driver, and then choose **Finish**.
6. Complete the Easysoft ODBC-Oracle Driver configuration dialog box.  
To find out how to do this, refer to the Connection attributes section.
7. To test the connection to Oracle, choose **Test**.  
Note that this doesn't test that the Easysoft ODBC-Oracle Driver is licensed. If you haven't yet [licensed](#) the Easysoft ODBC-Oracle Driver, this ODBC data source won't work with your application, even if the **Test** button succeeds.

## Connection attributes

- [Setting on Linux and UNIX](#)
- [Setting on Windows](#)

### Setting on Linux and UNIX

Your Easysoft ODBC-Oracle Driver data source in `odbc.ini` must contain these attributes if you're using the Instant Client with the Easysoft ODBC-Oracle Driver:

- The Oracle database user name (User).
- The Oracle database password (Password).
- The SQL connect URL string (Database).

-Or-

If you're using the standard Oracle client:

- The `tnsnames.ora` service name for the database (Database).

Here's an Instant Client example:

```
[demo-oracle]
Driver       = ORACLE
Database     = //testhost:1521/testdb
User        = system
Password     = p455w0rd
```

Here's a standard client example:

```
[demo-oracle]
Driver       = ORACLE
Database     = testdb
User        = system
Password     = p455w0rd
```

For more information about these mandatory attributes and other optional attributes, refer to this table:

Name	Value
DSN	The name of the data source. You'll need to specify this in your application. For example, your application may prompt you to choose this from a list of DSNs.
Description	Some applications display this to help users identify a particular data source.
Database	<p>If you're using the Instant Client, a SQL connect URL string. Use the following format:</p> <pre>//host:port/service-name</pre> <p>where <i>host</i> is the fully qualified domain name or IP address of the server on which the Oracle database is installed, <i>port</i> is the Oracle listener port or the alias name mapped to the port in the <code>/etc/services</code> file and <i>service-name</i> is the local net service name. For example, <code>//my_host:1521/XE</code>.</p> <p>If you're using the standard Oracle Client (or the Instant Client with a <code>tnsnames.ora</code> file by setting <code>TNS_ADMIN</code>), the logical name used to identify the Oracle target database. This is the local net service name defined in your <code>tnsnames.ora</code> file. For example, <code>my_database</code>.</p>
User	The name of the user that's supplied to Oracle to authenticate the connection.
Password	<p>The password supplied to Oracle to authenticate the connection.</p> <p>Note that passwords are case sensitive for new or modified accounts in Oracle 11 g and later.</p>
MetaData_ID	<p>When turned on (set to 1), the default value of the connection attribute <code>SQL_ATTR_METADATA_ID</code> is set to <code>SQL_TRUE</code>.</p> <p>If <code>SQL_TRUE</code>, the string arguments of catalog functions are treated as identifiers. The case is not significant. For non-delimited strings, the driver removes any trailing spaces, and the string is folded to uppercase. For delimited strings, the driver removes leading and trailing spaces, and takes literally whatever is between the delimiters.</p> <div style="border: 1px solid black; padding: 10px; margin: 10px 0;"> <p><b>Note</b> This attribute was introduced as a workaround for an issue in StarOffice. Setting this attribute can cause failures in applications that expect the default for <code>SQL_ATTR_METADATA_ID</code> to be <code>SQL_FALSE</code>.</p> </div> <p>By default, <code>Metadata_Id</code> is turned off.</p>
Metadata_Dont_Change_Case	<p>When turned on (set to 1), the Easysoft ODBC-Oracle Driver preserves the case of parameter values passed to metadata calls.</p> <p>By default, <code>Metadata_Dont_Change_Case</code> is turned off.</p>

Name	Value
VarcharTrimTrailingSpaces	<p>When turned on (set to 1), the Easysoft ODBC-Oracle Driver trims trailing spaces from VARCHAR types when passed as bound parameters. If VarcharTrimTrailingSpaces is set to 1, trailing spaces are removed from the end of the data.</p> <p>The default behaviour is to not trim spaces.</p>
Metadata_Dont_Do_Schema	<p>When turned on (set to 1), schema names are not returned by metadata calls. This is a workaround for some applications that do not handle schema names properly.</p> <p>By default, Metadata_Dont_Do_Schema is turned off.</p>
Use_Longs	<p>When turned on (set to 1), information on LONG data types is returned in the result set from a <a href="#">SQLGetTypeInfo</a> function call.</p> <p>Restrictions with LONG data types in Oracle databases (such as only permitting one column per table to be defined) often cause errors to occur, and this attribute can be used to include LONG within the list of valid data types that can be used by an application.</p> <p>The default for Use_Longs is off.</p>
Enable_Synonyms	<p>When turned on (set to 1), the Easysoft ODBC-Oracle Driver returns private synonyms for table names in metadata result sets.</p> <p>By default, the driver return synonyms. If you don't want synonyms, turn off Enable Synonyms. Including synonyms in metadata calls may greatly increase the size of metadata result sets for ODBC API calls such as <a href="#">SQLTables</a>.</p>
Enable_User_Catalog	<p>When turned on, the Easysoft ODBC-Oracle Driver returns metadata for the current Oracle user only. Turning on Enable_User_Catalog reduces the number of rows returned by <a href="#">SQLTables</a> calls.</p> <p>When turned off, the driver returns metadata for all users. Note that many ODBC applications will never need this amount of catalog data.</p> <p>By default, Enable_User_Catalog is turned on.</p>
Describe_Param_As_Strings	<p>Oracle doesn't support describing parameters, so the Easysoft ODBC-Oracle Driver doesn't support the <a href="#">SQLDescribeParam</a> ODBC call. However, if Describe_Param_As_Strings is turned on, the Easysoft ODBC-Oracle Driver describes any parameters as VARCHARs.</p> <p>By default, Describe_Param_As_Strings is turned off.</p>
DBI_Long_Size	<p>Any number you specify overrides the maximum size of a LONG column (in bytes).</p> <p>Perl DBI tries to allocate a buffer the size of a LONG column and, as this is rather large, it can cause problems, which setting DBI_LONG_SIZE can resolve.</p>



Name	Value
Connect_SQL	The SQL statement to run immediately after the Easysoft ODBC-Oracle Driver connects to the database.
No_LOBS	<p>When turned on (set to 1), No_LOBS increases the performance of the Easysoft ODBC-Oracle Driver if there are no CLOB or BLOB data types in use. This is only applicable to Oracle version 8.1.7.</p> <p>By default, No_LOBS is turned off.</p>
No_Parse	When turned on (set to 1), No_Parse prevents the Easysoft ODBC-Oracle Driver from preparing SQL (passed to <a href="#">SQLPrepare</a> and <a href="#">SQLExecDirect</a> ) to convert ODBC escapes and parameter markers. Doing this results in a small speed increase but prevents your application from using ODBC escapes sequences and parameter markers.
OCI_Attr_Prefetch_Rows	<p>The number of rows returned from a single "fetch" call made to the server.</p> <p>For example, if OCI_Attr_Prefetch_Rows is set to 10 then 10 rows are fetched from the database server. The next call to <a href="#">SQLFetch</a> doesn't need to make a call to the server as the required rows are held by the client already. The default value is 10. Increasing this value can reduce the number of round trip network calls to the server needed to return result sets at the expense of greater memory use.</p>
OCI_Attr_Prefetch_Memory	<p>The number of bytes of memory used on the client to store records returned from a single <a href="#">SQLFetch</a> call.</p> <p>This controls the number of records returned, which will be the total required to fill the allocated memory area.</p> <p>For example, if the available memory can store two rows then the next call to <a href="#">SQLFetch</a> doesn't need to make a call to the server, as the required row is held by the client already.</p>
Stmt_Caching	This attribute enables Oracle statement caching. Oracle statement caching establishes and manages a cache of statements within a session. It improves performance by efficiently using prepared cursors on the Oracle server and eliminating repetitive statement parsing. To enable caching, set this attribute to the size of the required cache. The attribute value should specify the number of statements to cache. Setting the attribute to 0 turns statement caching off. For more information about Oracle statement caching, refer to your Oracle documentation. The default is no statement caching.

Name	Value
Fake_CLOB_Length	<p>When connecting to Oracle 10 g or later from a Linux or UNIX platform, the Easysoft ODBC-Oracle Driver reports the length of BLOB, BFILE, and CLOB data types as 0. The driver does this because for these versions of Oracle, the maximum LOB size is 128 terabytes, which is too large a length for the ODBC API to handle.</p> <p>To change this default behaviour, set Fake_CLOB_Length to 1. When turned on (set to 1), the Easysoft ODBC-Oracle Driver sets the length to the largest value that the integer used to report the length is capable of holding. (Note that this is the default behaviour for the Easysoft ODBC-Oracle Driver on Windows, which is not affected by FAKE_CLOB_LENGTH.)</p> <p>By default, Fake_CLOB_Length is turned off.</p>
Pull_Lobs_Locally	<p>If your application crashes when selecting from multiple CLOB columns, try adding:</p> <pre data-bbox="469 828 1477 893">Pull_LOBS_Locally=1</pre> <p>to your ODBC data source or connection string. When set to 1, the Easysoft ODBC-Oracle Driver returns the entire contents of the CLOB data to an internally allocated buffer. This lets the Easysoft ODBC-Oracle Driver determine the byte length of the CLOB. This avoids the application from being given a short length when the character length for the data is not the same as the byte length.</p> <p>By default, Pull_LOBS_Locally is turned off (set to 0).</p>
OCI_UTF_Flag	<p>When turned on (set to 1), the Easysoft ODBC-Oracle Driver does additional conversion when reading LOB data. The Easysoft ODBC-Oracle Driver does this to compensate for non-conformant OCILobRead behaviour when reading multibyte character data. When turned off (set to 0), the Easysoft ODBC-Oracle Driver assumes that the OCILobRead behaviour conforms to the Oracle documentation.</p> <p>Setting OCI_UTF_Flag to 1 may provide a workaround if you experience problems when reading UTF-8 LOB data in parts (that is, the buffer size passed to <a href="#">SQLGetData</a> is not large enough to hold the entire LOB) and you are using the Instant Client 11.1 or later.</p> <p>By default, OCI_UTF_Flag is turned off.</p>

Name	Value
With_Unicode	<p>When turned on (set to 1), the Easysoft ODBC-Oracle Driver attempts to detect whether the national character set for the current environment is AL16UTF16. If this is the case, the Easysoft ODBC-Oracle Driver:</p> <ul style="list-style-type: none"> <li>• Describes NCHAR, NVARCHAR2, or NCLOB columns as SQL_WCHAR, SQL_WVARCHAR, and SQL_WLONGVARCHAR, when describing columns in a result set.</li> <li>• Transfers data as 16-bit Unicode when binding parameters, if the SQL data type is SQL_WCHAR, SQL_WVARCHAR, or SQL_WLONGVARCHAR.</li> </ul> <p>If the column type is one of the above, the column is bound to the implementation row descriptor (IRD) expecting the length returned and bound size to be in units of twin bytes.</p> <p>To check what national character set the Easysoft ODBC-Oracle Driver has detected, set With_Unicode to 1, turn on Easysoft ODBC-Oracle Driver logging by adding the entry:</p> <pre>LOG = /tmp/oracle.log</pre> <p>to your data source, and then run a query against a table containing a NCHAR, NVARCHAR2, or NCLOB column. Look in the log file for text similar to:</p> <pre>Looking at column of type 1 with charset_id of 2000 against al16utf16_csid = 2000</pre> <p>If the charset_id and al16utf16_csid values don't match, setting With_Unicode will have no effect.</p> <p>By default, With_Unicode is turned off.</p>
Pool_Type	The type of pooling required. This can be SESSION or CONNECTION.
Pool_Scope	This can be ENV or GLOBAL. This either associates the pool with the ODBC environment or makes it a global resource.
Pool_Initial	The number of sessions or connections that are created when the pool is created.
Pool_Max	The maximum number of sessions or connections that the pool can contain.
Pool_Increment	The number that the session or connection count is incremented by.
Pool_Username	The database user name with which to authenticate the sessions or connections.
Pool_Password	The password for Pool_Username.
Pool_Database	This database for which the pools are created.

Name	Value
Pool_Connection_Class	<p>Database Resident Connection Pooling (DRCP) guarantees that pooled servers are never shared across different users. Setting Pool_Connection_Class allows for further separation between the sessions of a given user by defining a connection class. A connection class lets different applications (connecting as the same database user) identify their sessions using a logical name that corresponds to the application. OCI then ensures that sessions belonging to a particular connection class are not shared outside of the connection class.</p> <p>OCI supports a maximum connection class length of 1024 bytes. The asterisk character (*) is a special character and is not allowed in the connection class name.</p>
Pool_Purity	<p>Whether the application requests a brand new session or reuses a session from the DRCP pool.</p> <p>To request a new session, set Pool_Purity to NEW. The other value Pool_Purity supports is SELF.</p> <p>If you connect to a DRCP-enabled database server without setting Pool_Purity, sessions are reused. When reusing a session from the pool, the NLS attributes of the server take precedence over that of the client.</p>
XA_Connection_String	<p>The name of the database specified with the DB field in the xa_open string. For example, you specify a database named payroll with the following xa_open string clause:</p> <pre>DB=payroll</pre> <p>You also need to specify payroll as the value for the XA_Connection_String attribute:</p> <pre>XA_Connection_String=payroll</pre> <p>XA_Connection_String is only necessary if you're using the Easysoft ODBC-Oracle Driver to connect to Oracle in the context of an XA transaction and the Transaction Manager specifies a named database in the xa_open string.</p>
XaoswName	Sets the XA entry point for the Easysoft ODBC-Oracle Driver. Note that this is set in the odbcinst.ini file, which is normally located in /etc. The default value is xaosw.
Data_Type_Map	<p>How the Easysoft ODBC-Oracle Driver maps Oracle data types onto ODBC data types. The available maps are NUMERIC (set the attribute to 0), DOUBLE (set the attribute to 1), BIGINT+DOUBLE (set the attribute to 2), and BIGINT+NUMERIC (set the attribute to 3).</p> <p>The default for Data_Type_Map is 0.</p>

## Data type maps

Value	Oracle data type	ODBC data type
0	NUMBER $\leq$ 4 digits	SQL_SMALLINT
	NUMBER $\leq$ 9 digits	SQL_INTEGER
	NUMBER = $n$ digits	SQL_NUMERIC
	NUMBER = $n,m$ digits	SQL_NUMERIC
1	NUMBER $\leq$ 4 digits	SQL_SMALLINT
	NUMBER $\leq$ 9 digits	SQL_INTEGER
	NUMBER = $n$ digits	SQL_DOUBLE
	NUMBER = $n,m$ digits	SQL_DOUBLE
2	NUMBER $\leq$ 4 digits	SQL_SMALLINT
	NUMBER = $n$ digits	SQL_INTEGER
	NUMBER $\leq$ 19 digits	SQL_BIGINT
	NUMBER $n,m$ digits	SQL_DOUBLE
3	NUMBER $\leq$ 4 digits	SQL_SMALLINT
	NUMBER $\leq$ 9 digits	SQL_INTEGER
	NUMBER $\leq$ 19 digits	SQL_BIGINT
	NUMBER >9 digits	SQL_NUMERIC

## Setting on Windows

The Easysoft ODBC-Oracle Driver data source configuration dialog box, accessible when you create or edit an Easysoft ODBC-Oracle Driver data source in **ODBC Data Source Administrator** lets you configure your data source.

For information about the data source attribute fields the dialog box contains, refer to [this topic](#).

## 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-Oracle Driver connection string:

```
DRIVER=ORACLE;DB=my_database;UID=system;PWD=p455w0rd;
```

For a list of the other attributes you can set in the connection string, refer to [this section](#).

## Logging

If you report an issue to us, we may ask you to turn on ODBC Driver Manager or Easysoft ODBC-Oracle 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-Oracle 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-Oracle Driver logging is specific to the Easysoft driver and is of most use when making a support call.

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

```
[demo-oracle]
.
.
.
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.

## ODBC Driver Manager logging on Windows

1. In the Windows **taskbar search** box, enter "Run".
2. Do one of the following:
  - If your application is 64-bit, in the **Run** dialog box, enter:

```
odbcad32.exe
```

-Or-

- If your application is 32-bit, in the **Run** dialog box, enter:

```
%windir%\syswow64\odbcad32.exe
```

**Note**

If you're not sure whether your application is 32-bit or 64-bit, start your application, then in Windows **Task Manager** check whether your application's process name contains (32-bit). For example, the process name for the 32-bit version of Excel is Microsoft Excel (32-bit); the process name for the 64-bit version of Excel is Microsoft Excel. On older versions of Windows, 32-bit applications contain \*32 in the process name rather than (32-bit).

For applications such as Oracle or SQL Server that run as a service, check the \*Background processes\* list rather than the **Apps** list in **Task Manager**. If you're running a programming language from within a Windows command-line shell (for example, Command or PowerShell), in your shell, run the .exe file for the programming language. For example, run perl, php, python, or node. In **Task Manager**, expand the process list for **Windows Command Processor** or **Windows PowerShell**, as appropriate, and check whether the process for your programming language contains (32-bit).

3. Choose the **Tracing** tab.
4. Select **Machine-Wide tracing for all identities**.
5. Enter a log file name and path in the space provided. For example:

```
C:\Windows\Temp\SQL.log
```

6. Choose **Start Tracing Now**.



**Note**

With SQL Server, you may get two Driver Manager log files, we need both. The first log file is in the folder that you specify in **ODBC Data Source Administrator**. The second file's location is defined by SQL Server. Two possible locations are the top-level folder (for example, C:\SQL.log) or the SQL Server temporary folder (for example, C:\Users\MSSQL\$SQLEXPRESS\AppData\Local\Temp\SQL.log). If the Driver Manager log file isn't in these folders, search for it on the drive where SQL Server is installed.

## Finding out what product version you have on Windows

If you have an issue with the Easysoft ODBC-Oracle Driver, we may ask you to tell us what your product version is. To find this out:

1. In the Windows **taskbar**, enter “Add or remove programs” in the Windows **search** box.
2. Select Easysoft ODBC-Oracle Driver in the list.

The product version displays below.

## Client applications

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

- [Microsoft Access](#)
- [Microsoft Excel](#)
- [Microsoft Power BI](#)
- [SQL Server](#)
- [LibreOffice](#)
- [Go](#)
- [Node.js](#)
- [Perl](#)
- [PHP](#)
- [Python](#)
- [R](#)

## Microsoft Access

1. [Install the Easysoft ODBC-Oracle Driver](#) on same computer as Access.
2. [Configure an ODBC data source](#).
3. Choose one of the following ways to work with your Oracle data in Access.

### Linking a table

1. Open your Microsoft Access database.
2. Choose **External Data**.
3. In the **New Data Source** list, choose **From Other Sources > ODBC Database**.
4. In the **Get External Data** screen, choose **Link to the data source by creating a linked table**, and choose **OK**.
5. In the **Select Data Source** dialog box, choose the **Machine Data Source** tab.
6. Choose your Easysoft ODBC-Oracle Driver ODBC data source from the **Machine Data Source** list, and then choose **OK**.
7. In the **Link Tables** dialog box, choose the tables that you want to link to, and then choose **OK**.

### Importing a table

1. Open your Microsoft Access database.
2. Choose **External Data**.
3. In the **New Data Source** list, choose **From Other Sources > ODBC Database**.
4. In the **Get External Data** screen, choose **Import the source data into a new table in the current database**, and choose **OK**.
5. In the **Select Data Source** dialog box, choose the **Machine Data Source** tab.
6. Choose your Easysoft ODBC-Oracle Driver ODBC data source from the **Machine Data Source** list, and then choose **OK**.
7. In the **Import Objects** dialog box, choose the tables you want to import, and then choose **OK**.

## Microsoft Excel

1. [Install the Easysoft ODBC-Oracle Driver](#) on same computer as Excel.
2. [Configure an ODBC data source](#).
3. Choose one of the following ways to work with your Oracle data in Excel.

## Data Connection Wizard

1. Choose **Data > Get Data > From Other Sources > From ODBC**.
2. Choose your Easysoft ODBC-Oracle Driver data source from the list, and then choose **OK**.
3. Enter the user name and password for your data store if applicable, otherwise, enter any text string to get past this stage. Choose **Next**.
4. Choose the table that contains the data you want to retrieve, and then choose **Load**.

## Microsoft Query

1. Choose **Data > Get Data > From Other Sources > From Microsoft Query**.
2. In the **Choose Data Source** dialog box, choose your Oracle data source from the list, and then choose **OK**.
3. In the **Query Wizard**, choose the columns that contain the data you want to retrieve, and then click **Next**.
4. If you want to return a subset of the data, use the **Filter Data** screen to filter the results of your query (this is the equivalent of a SQL WHERE clause), and then choose **Next**.
5. If you want to change the sort order of your data, use the **Sort Order** screen to sort the results of your query (this is the equivalent of a SQL ORDER BY clause), and then choose **Next**. Choose **Finish** to return your Oracle data to Excel.

## PowerPivot

1. On the **PowerPivot** tab, choose **Manage**.
2. In the **PowerPivot** window, choose **Get External Data > From Other Sources**.
3. In the **Connect to a Data Source** list, choose **Others (OLEDB/ODBC)**.
4. In the **Specify a Connection** screen, enter a name for your connection in the space provided. Then choose **Build**.
5. In the **Data Link Properties** box, choose your Easysoft ODBC-Oracle Driver data source from the list, and then choose **OK**.
6. Choose **Next**.
7. Choose how to import your Oracle data and then choose **Finish**.
8. Choose **Close** to return the data to Excel.

## Microsoft Power BI

1. [Install the Easysoft ODBC-Oracle Driver](#) on same computer as Power BI Desktop.
2. [Configure an ODBC data source](#).
3. In Power BI Desktop, choose **Get data from another source**.
4. In the **Get Data** dialog box, choose **ODBC**, and then choose **Connect**.
5. In the **From ODBC** dialog box, choose your Oracle data source, and then choose **OK**.
6. Enter your database user name and password when prompted.

If you make a mistake when entering the user name and password, cancel the connection process. Then in Power BI Desktop **Options and Settings**, edit the data source. Specify the correct user name or password in the data source credentials dialog box. Otherwise, Power BI Desktop will continue to use the cached incorrect credentials.

<b>Note</b>	If you do not normally need to enter a user name and password, enter some dummy strings in the spaces provided.
-------------	---

7. In the **Navigator** dialog box, choose the tables you want to analyse in Power BI Desktop, and then choose **Load**.

Your Oracle data is now available to use in Power BI visualisations.

## Microsoft SQL Server

1. [Install the Easysoft ODBC-Oracle Driver](#) on same computer as SQL Server.
2. [Configure an ODBC data source](#).
3. In Microsoft SQL Server Management Studio, connect to the SQL Server instance you want to create the linked server against.

You need to log on with an account that is a member of the SQL Server sysadmin fixed server role to create a linked server.

4. Right-click **Server Objects**. From the pop-up menu choose **New > Linked Server**.
5. In the **Linked server** box, enter "Oracle".
6. From the **Provider** list, choose **Microsoft OLE DB Provider for ODBC drivers**.
7. In the **Data source** box, enter the name of your Oracle data source, and then choose **OK**.

SQL Server verifies the linked server by testing the connection.

- If you get the error "Specified driver could not be loaded due to system error 126: The specified module could not be found," choose **Yes** when prompted whether to keep the linked server. You need to restart your SQL Server instance before you can use the linked server. If SQL Server was already running when you installed the Easysoft ODBC-Oracle Driver, it will not have the latest version of the System Path environment variable. The Easysoft ODBC-Oracle Driver Setup program adds entries for the driver to the System Path. Restarting the instance makes these changes available to SQL Server, allowing it to load the Easysoft ODBC-Oracle Driver.
  - If you made a mistake when specifying the Easysoft ODBC-Oracle Driver, you get the error "Data source name not found and no default driver specified." If you get this error, choose **No** when prompted whether to keep the linked server and edit the value in the **Data source** box.
8. You can query your Easysoft ODBC-Oracle Driver data either by using a:
    - Four part table name in a distributed query.

A four part table name has the format:

```
server_name.[database_name].[schema_name].table_name
```

For data stores where there is no database or schema, Easysoft ODBC drivers return a "dummy" value for both identifiers, because some ODBC applications expect there to be a database and a schema. To find out the identifier names, run:

```
EXEC sp_tables_ex @table_server = 'Oracle'
```

If present, include these identifiers in your SQL statements. If not present, omit them. For example:

```
SELECT * FROM [Oracle]..DBO.Customers
```

The capitalisation of the table name must be the same as it is in the result set returned by sp\_tables\_ex.

- Pass-through query in an OPENQUERY function. For example:

```
SELECT * FROM OPENQUERY([Oracle], 'SELECT * FROM Customers')
```

```
-- If you get an "RPC not enabled for this server" message, right-click your
-- linked server and choose Properties.
```

```
-- In Server Options, set both RPC and RPC Out to `True`.
```

```
EXEC ('INSERT INTO Customers (Surname, GivenName, City, Phone, CompanyName)
VALUES ('Devlin' , 'Michaels' , 'Kingston' , '2015558966' ,
'PowerGroup'))
AT Oracle

UPDATE OPENQUERY ([Oracle], 'SELECT Surname FROM Customers WHERE CompanyName =
'PowerGroup') SET Surname='Jones'
DELETE OPENQUERY (Oracle, 'SELECT Surname FROM Customers WHERE CompanyName =
'PowerGroup')
```

SQL Server sends pass-through queries as uninterpreted query strings to the Oracle. This means that SQL Server does not apply any kind of logic to the query or try to estimate what that query will do.



## LibreOffice

1. [Install the Easysoft ODBC-Oracle 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-Oracle 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 Oracle 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=Oracle")
    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-Oracle 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 Oracle data:

```
const odbc = require('odbc');
// Replace Oracle with the name of your Easysoft ODBC-Oracle Driver
// data source.
const connection = odbc.connect('DSN=Oracle', (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-Oracle Driver data source:

```
const odbc = require('odbc');
const connection = odbc.connect('DSN=Oracle', (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=Oracle', (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 Oracle data:

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

```
)
VALUES
(
  'Devlin',
  'Michaels',
  'Kingston',
  '2015558966',
  'PowerGroup'
)", (error, result) => {
  if (error) { console.error(error) }
  console.log(result);
});

});

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

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

## Perl

1. [Install the Easysoft ODBC-Oracle 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-Oracle Driver to connect to Oracle.

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

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

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-Oracle Driver data source:

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

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:Oracle');

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 Oracle data:

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

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

$sth->execute('Devlin', 'Michaels', 'Kingston', '2015558966', 'PowerGroup');

$dbh->disconnect if ($dbh);

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

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

$sth->execute('PowerGroup');

$dbh->disconnect if ($dbh);

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

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

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

### Further information

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

## PHP

1. [Install the Easysoft ODBC-Oracle 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 Oracle data:

```
<?php
// Replace Oracle with the name of your Easysoft ODBC-Oracle Driver data
source.
// If your database requires a user name and password, supply them in the
odbc_connect_call.
$con = odbc_connect("Oracle", "", "");
$stmt = odbc_exec($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-Oracle Driver data source:

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

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

```
<?php
$cnx = odbc_connect("Oracle", "", "");
$stmt = odbc_prepare($cnx, "INSERT INTO Customers (Surname, GivenName, City,
Phone, CompanyName) VALUES (?, ?, ?, ?, ?)");
```



```
$success = odbc_execute($stmt, array('Devlin', 'Michaels', 'Kingston',  
'2015558966', 'PowerGroup'));  
odbc_close($cnx);  
?>  
  
<?php  
$cnx = odbc_connect("Oracle", "", "");  
$stmt = odbc_prepare($cnx, "UPDATE Customers SET Surname = 'Jones' WHERE  
CompanyName = ?");  
$success = odbc_execute($stmt, array('PowerGroup'));  
odbc_close($cnx);  
?>  
  
<?php  
$cnx = odbc_connect("Oracle", "", "");  
$stmt = odbc_prepare($cnx, "DELETE FROM Customers WHERE CompanyName = ?");  
$success = odbc_execute($stmt, array('PowerGroup'));  
odbc_close($cnx);  
?>
```

### Further information

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

## Python

1. [Install the Easysoft ODBC-Oracle 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 Oracle data:

```
import pyodbc

# Replace Oracle with the name of your Easysoft ODBC-Oracle Driver data source.
cnxn = pyodbc.connect("DSN=Oracle")
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-Oracle Driver data source:

```
import pyodbc

# Replace Oracle with the name of your Easysoft ODBC-Oracle Driver data source.
cnxn = pyodbc.connect("DSN=Oracle")
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 Oracle with the name of your Easysoft ODBC-Oracle Driver data source.
```

```
cnxn = pyodbc.connect("DSN=Oracle")
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 Oracle data:

```
import pyodbc

cnxn = pyodbc.connect("DSN=Oracle")
cursor = cnxn.cursor()
sql = "INSERT INTO Customers (Surname, GivenName, City, Phone, CompanyName)
VALUES (?, ?, ?, ?, ?)"
cursor.execute(sql, 'Devlin', 'Michaels', 'Kingston', '2015558966', 'PowerGroup')
cursor.commit()
exit()
```

```
import pyodbc

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

```
import pyodbc

cnxn = pyodbc.connect("DSN=Oracle")
cursor = cnxn.cursor()
sql = "DELETE FROM Customers WHERE CompanyName = ?"
cursor.execute(sql, 'PowerGroup')
cursor.commit()
exit()
```

## Further information

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

## R

1. [Install the Easysoft ODBC-Oracle 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 Oracle data:

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

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

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

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

```
library("RODBC")
# Replace Oracle with the name of your Easysoft ODBC-Oracle Driver data source.
ch <- odbcConnect("Oracle")
# 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 Oracle data:

```
library("RODBC")
ch <- odbcConnect("Oracle")
sqlQuery(ch, paste("INSERT INTO Customers (Surname, GivenName, City, Phone,
CompanyName) VALUES ('Devlin', 'Michaels', 'Kingston', '2015558966',
'PowerGroup')"))
odbcClose(ch)
quit()
```

```
library("RODBC")
ch <- odbcConnect("Oracle")
sqlQuery(ch, paste("UPDATE Customers SET Surname = 'Jones' WHERE CompanyName = 'PowerGroup'"))
odbcClose(ch)
quit()

library("RODBC")
ch <- odbcConnect("Oracle")
sqlQuery(ch, paste("DELETE FROM Customers WHERE CompanyName = 'PowerGroup'"))
odbcClose(ch)
quit()
```

## About the Easysoft ODBC-Oracle Driver

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

In this section:

- [ODBC API and scalar functions](#)
- [Cursor support](#)
- [Advanced Security](#)
- [Oracle Real Application Clusters \(RAC\)](#)
- [Transparent Application Failover \(TAF\)](#)
- [Database Resident Connection Pooling \(DRCP\)](#)
- [Network Protocols](#)
- [Materialized Views](#)
- [XA support](#)
- [Supported data types](#)
- [Example SQL statements](#)

## ODBC API and scalar functions

### API functions

The Easysoft ODBC-Oracle Driver supports all ODBC 3.x functions apart from:

- [SQLDescribeParam](#)

For more information, refer to the `Describe_Param_As_Strings` attribute description in the [connection attributes section](#).

The Easysoft ODBC-Oracle Driver partially supports [SQLSetPos](#). An application can use the Easysoft ODBC-Oracle Driver to specify a cursor position by calling `SQLSetPos` with the `SQL_POSITION` argument.

### Scalar functions

The Easysoft ODBC-Oracle Driver supports all [scalar](#) functions apart from:

- `BIT_LENGTH`
- `CEILING`
- `CHAR`
- `CHARACTER_LENGTH`
- `CHAR_LENGTH`
- `LCASE`
- `OCTET_LENGTH`
- `UCASE`

Use the ODBC syntax with scalar functions. For example:

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

## Cursor support

The Easysoft ODBC-Oracle Driver supports FORWARD\_ONLY and STATIC cursors.



## Advanced Security

The Oracle Advanced Security option is an Oracle client or server add-on that combines network encryption, database encryption, and strong authentication to protect sensitive data stored in Oracle databases. The Advanced Security option:

- Guarantees data integrity by detecting whether it has been modified during transmission.
- Encrypts data using encryption standards such as RSA or DES to ensure data privacy.
- Supports third party authentication services such as Kerberos and RADIUS.

Oracle Advanced Security requires Net8 or Oracle Net to transmit data securely. The Easysoft ODBC-Oracle Driver uses the Oracle client, which uses these protocols to communicate with the Oracle database server over the network. Applications that access Oracle by using the Easysoft ODBC-Oracle Driver can therefore take advantage of the Oracle Advanced Security option.

For information about configuring the Oracle client and server for use with the Oracle Advanced Security option, refer to the Oracle Advanced Security Administrator's Guide. No additional Easysoft ODBC-Oracle Driver configuration is necessary.

## Oracle Real Application Clusters (RAC)

Real Application Clusters (RAC) is an Oracle High Availability feature that enables an Oracle Database Server Grid by providing a single database that spans multiple low-cost servers yet appears to the application as a single, unified database system. RAC combines the processing power of these multiple interconnected computers to provide system redundancy, scalability, and high availability. Application scale in an RAC environment to meet increasing data processing demands without changing the application code.

To increase the performance of a RAC database, you can add cluster nodes. Each additional node can help speed up application processing.

If a clustered server fails, the Oracle database will continue running on the surviving servers. If more processing power is needed, another server can be added without interrupting user's access to data.

The Easysoft ODBC-Oracle Driver can access Oracle RAC environments. The Oracle client must be configured correctly to take full advantage of the RAC features. Please consult you Oracle documentation for more details.

## Transparent Application Failover (TAF)

Transparent Application Failover (TAF) is a mechanism that automatically reconnects client applications to a node of a RAC database cluster following a connection failure. If a failure occurs, the Oracle client intercepts the resultant error message and starts the transparent failover process. The Oracle client requests another connection from the Oracle listener, which then connects the client to a surviving node of the RAC database cluster.

There may be a delay associated with failing over to another node. To keep users informed, it is possible for an OCI application to register a callback function that is invoked in the event of a connection loss and during the course of the failover. The callback function enables the OCI application to advise users that a failover is in progress and to wait while the failover completes. This ensures users do not attempt to restart their applications, because they perceive this delay as a application failure.

The Easysoft ODBC-Oracle Driver enables ODBC applications to register a failover callback function. To do this, the ODBC application must:

1. Define a callback function that takes the form:

```
int TAF_callback_fn( SQLHANDLE connection,
    int type,
    int event );
```

where:

- *connection* is the ODBC connection handle.
- *type* is the Oracle failover type, which tells the callback function what type of failover the client has requested. The failover types are OCI\_FO\_NONE, OCI\_FO\_SESSION, OCI\_FO\_SELECT, and OCI\_FO\_TXNAL.
- *event* is the type of Oracle failover event that took place, which tells the callback function why it was called. The failover events are OCI\_FO\_END, OCI\_FO\_ABORT, OCI\_FO\_REAUTH, OCI\_FO\_BEGIN, and OCI\_FO\_ERROR.

The function can trigger a new failover attempt by returning OCI\_FO\_RETRY.

All the OCI\_\* constants listed here are defined in the OCI header file oci.h. Consult the Oracle Call Interface Programmer's Guide for the meanings of these constants.

2. Register and establish a context for the callback function by calling [SQLSetConnectAttr](#) with the attributes SQL\_ATTR\_REGISTER\_TAF\_HANDLE and SQL\_ATTR\_REGISTER\_TAF\_CALLBACK.

The definitions for these attributes are:

```
#define SQL_ATTR_REGISTER_TAF_CALLBACK 1280
#define SQL_ATTR_REGISTER_TAF_HANDLE 1281
```

The value SQL\_ATTR\_REGISTER\_TAF\_CALLBACK is a pointer to the callback function. The value for SQL\_ATTR\_REGISTER\_TAF\_HANDLE is a pointer to the connection handle used to establish a context for the callback function. For example:

```
SQLSetConnectAttr(dbc,
    1280 /* SQL_ATTR_REGISTER_TAF_CALLBACK */,
    &TAF_callback_fn,
    SQL_IS_POINTER);

SQLSetConnectAttr(dbc,
```

```
1281 /*SQL_ATTR_REGISTER_TAF_HANDLE*/ ,  
&dbc ,  
SQL_IS_POINTER) ;
```

Version 1.39 of the Perl DBD::ODBC module (which combined with Perl DBI provides an interface to ODBC databases for Perl) has been used to test the Easysoft ODBC-Oracle Driver's TAF support. An example Perl script is provided that shows how to use the failover types and events to:

- Keep the user informed throughout the duration of the failover.
- Abort the failover.

An example Perl script is available at:

[https://search.cpan.org/~mjevans/DBD-ODBC-1.39/ODBC.pm#odbc\\_taf\\_callback](https://search.cpan.org/~mjevans/DBD-ODBC-1.39/ODBC.pm#odbc_taf_callback)

## Database Resident Connection Pooling (DRCP)

Database Resident Connection Pooling (DRCP) is a scalability feature introduced in Oracle 11g Release 1, which uses a combination of dedicated server and connection broker to handle short, transient sessions coming from Web applications.

DRCP is especially relevant for architectures with multi-process, single-threaded application servers, such as PHP and Apache, that cannot do middle tier connection pooling.

The OCI client libraries enable applications to configure the behaviour of DRCP. Applications can:

- Request a brand new session if they cannot reuse a session from the pool.
- Specify a connection class that indicates that the application is willing to reuse a pooled server, which was used by other applications using the same connection class.

For example, applications in an HR suite may be willing to share pooled servers among themselves but not among other applications.

Because the Easysoft ODBC-Oracle Driver uses Oracle client software, it makes it possible for ODBC applications and interfaces to configure behaviour normally controlled from within the OCI layer. For example, the Easysoft ODBC-Oracle Driver enables PHP applications using the Unified ODBC interface to configure the DRCP pool purity or specify a connection class.

To configure the DRCP pool purity or specify a connection class from an ODBC application, set the `Pool_Purity` or `Pool_Connection_Class` Easysoft ODBC-Oracle Driver attributes. (Note that to set these attributes, you need to be using version 11.1+ of the Oracle client software.)

## Background

A connection pool is a cache of database connection objects. The objects represent physical database connections that can be used by an application to connect to a database.

Connection pools promote the reuse of connection objects and reduce the number of times that connection objects are created. Connection pools significantly improve performance for database intensive applications because creating connection objects is costly both in terms of time and resources.

The connection pool is normally configured with a shared pool of physical connections, translating to a back-end server pool containing an identical number of dedicated server processes.

Applications that can use connection pooling include middle tier applications for Web application servers and e-mail servers. (Web applications introduced the three-tier model in which the browser is the client tier, the database is the backend tier, and the web server and its extensions are the middle tier.)

Connection pooling is beneficial only if the middle tier itself is multithreaded, because it takes advantage of the ability of multiple threads in one application process to share resources. (Threads are lightweight processes that exist within a larger process.)

DRCP is an alternative connection pooling mechanism that enables multi-process applications to share connections to the database. (In multi-process applications, unlike in multi-threaded applications, processes are insulated from each other.)

## Network Protocols

The Easysoft ODBC-Oracle Driver supports both IPC and TCP/IP network protocols.

---

## Materialized Views

The Easysoft ODBC-Oracle Driver supports materialized views. A materialized view is a database object that contains the results of a query. Materialized views stored in the same database as their base tables can improve query performance through query rewrites.

The query rewrite mechanism reduces response time for returning results from the query. It does this by automatically rewriting the SQL query to use the materialized view instead of accessing the original tables. Query rewrites are particularly useful in a data warehouse environment.

For more information about materialized views, refer to your Oracle documentation.

## XA support

The Easysoft ODBC-Oracle Driver can be configured to take part in a distributed XA transaction. To do this, add a data source that contains entries that correspond with the xa\_open string used by the XA Transaction Manager to connect to the Oracle database server.

For example, the following sample xa\_open string opens an XA connection to an Oracle database named payroll. It also specifies the Net8 link and the username and password used to log onto the Oracle server.

```
Oracle_XA+sqlnet=ninetwo.oracle+SesTm=35+Acc=P/system/manager+Threads=true+DB=payroll
```

To make this XA connection available for use by the Easysoft ODBC-Oracle Driver, the data source used to access Oracle needs to contain the following corresponding entries:

```
[ORACLE-XA]
Driver           = ORACLE
Database         = ninetwo.oracle
User             = system
Password         = manager
XA_Enlist        = 1
XA_Connection_String = payroll
```

In addition, the XA\_Enlist attribute must be set to 1. When this setting is present, the Easysoft ODBC-Oracle Driver accesses Oracle by using the Oracle XA library. Note that the data source can then only be used to access an Oracle database as an XA resource under the control of a Transaction Manager. If you need to access the same database with a non-XA connection, configure a separate data source without the XA\_Enlist attribute.

The XA\_Connection\_String attribute is only required if the DB field is present in the xa\_open string. The XA\_Connection\_String attribute value must be the same as that of the DB field.

A tutorial that provides more information about using the Easysoft ODBC-Oracle Driver in the context of an XA transaction is available on the [Easysoft web site](#). The tutorial includes a code sample that shows the point at which an ODBC connection needs to be created and closed to participate in a distributed transaction.



## Supported data types

The Easysoft ODBC-Oracle Driver supports these Oracle data types:

- BLOB
- BFILE
- RAW
- CLOB
- CHAR
- NUMBER

To control how NUMBER data types are mapped onto ODBC data types, use the Data\_Type\_Map parameter. For more information, refer to the [connection attributes section](#).

- BINARY\_FLOAT
- BINARY\_DOUBLE
- DECIMAL
- INTEGER
- FLOAT
- DOUBLE PRECISION
- VARCHAR2
- TIMESTAMP
- TIMESTAMP WITH TIME ZONE
- TIMESTAMP WITH LOCAL TIME ZONE
- DATE

## Finding out more about data types on Windows

If you need more information about a data types, for example, the precision and scale, use Microsoft's ODBC Test to do this.

1. Download the version of ODBC Test that matches your application's architecture from:  
<https://www.easysoft.com/ftp/pub/utils/windows/odbc-test/>
2. Copy both files to a folder on the machine where Easysoft ODBC-Oracle Driver is installed.
3. Double-click **odbcte32.exe**.
4. Select **Con > Full Connect**.
5. Choose your Easysoft ODBC-Oracle Driver data source from the list.
6. Choose **Catalog > SQLGetTypeInfo**.
7. Either choose **SQL\_ALL\_TYPES=0 (1.0)** or a specific data type from the **DataType** list.
8. Choose **Results > Get Data All**.

## Example SQL statements

### 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
```

- 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 EXTRACT(YEAR FROM OrderDate) = 2025
```

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'])
```

## 68 Example queries

---

- 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
    );
```

- 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 values, use the SQL aggregate functions MIN(), MAX(), COUNT(), SUM(), and AVG(). For example:

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

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

```
SELECT CAST(Quantity AS Char(100))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
```

## 70 Example queries

---

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

```
SELECT City FROM Contacts
UNION
SELECT City FROM Customers
```

- 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
```

- 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 LIMIT. For example:

```
SELECT * FROM Customers LIMIT 10
```

- 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 Oracle record, use INSERT INTO. For example:

```
INSERT INTO
  Customers (
    Surname,
    GivenName,
    City,
    Phone,
    CompanyName
  )
VALUES
  (
    'Devlin',
    'Michaels',
    'Kingston',
    '2015558966',
    'PowerGroup'
  )
```

- Here's a SQL Server linked server example:

```
EXEC ('INSERT INTO Customers (Surname, GivenName, City, Phone, CompanyName)
VALUES ('Devlin' , 'Michaels' , 'Kingston' , '2015558966' ,
'PowerGroup'))
```



- Here's an Oracle linked table example:

```
DECLARE
    num_rows integer;
BEGIN
    num_rows:=DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE@Link
    ('INSERT INTO Customers (Surname, GivenName, City, Phone, CompanyName) VALUES
    ('Devlin', 'Michaels', 'Kingston', '2015558966', 'PowerGroup')');
END;
/
```

- The Easysoft ODBC-Oracle 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, Phone,
CompanyName) VALUES (?, ?, ?, ?, ?)/)
    or die "Can't prepare statement: $DBI::errstr";

$sth->execute('Devlin', 'Michaels', 'Kingston', '2015558966', 'PowerGroup');
```

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

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

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

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

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

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

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

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

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

# Index

## A

### Access

- importing Oracle data, [36](#)
- linking Oracle data, [36](#)

## B

### Base

- working with Oracle data, [41](#)

## C

- Connect\_SQL attribute, [25](#)
- connecting to Oracle, [19](#)
- connection string attributes, [30](#)
- cursor support, [56](#)

## D

- data source attributes, [22](#)
- data sources
  - removing
    - on Windows, [18](#)
- data types, [65](#)
- Data\_Type\_Map attribute, [28](#)
- Database attribute, [23](#)
- Database Resident Connection Pooling, [61](#)
- DBI\_Long\_Size attribute, [24](#)
- Describe\_Param\_As\_Strings attribute, [24](#)
- Description attribute, [23](#)
- DSN attribute, [23](#)
- DSN-less connections, [30](#)

## E

- Easysoft ODBC-Oracle Driver
  - adding ODBC data sources
    - on Linux or UNIX, [19](#)
    - on Windows, [21](#)
  - connecting to Oracle, [19](#)
  - cursor support, [56](#)
  - data source attributes, [22](#)
  - DSN-less connections, [30](#)
  - installing
    - on Linux or UNIX, [7](#)
    - on Windows, [16](#)
  - logging, [31](#)
  - ODBC API support, [55](#)
  - scalar function support, [55](#)
  - setting the environment for, [19](#)
  - SQL examples, [66](#)
  - supported data types, [65](#)
  - uninstalling
    - on Linux or UNIX, [15](#)
    - on Windows, [18](#)
  - XA support, [64](#)
- Enable\_Synonyms attribute, [24](#)

- Enable\_User\_Catalog attribute, [24](#)

- environment variables, [19](#)

### Excel

- importing Oracle data with Data Connection Wizard, [37](#)
- importing Oracle data with PowerPivot, [37](#)
- importing Oracle data with Query, [37](#)

## F

- Fake\_CLOB\_Length attribute, [26](#)

## G

### Go

- working with Oracle data, [42](#)

## I

- installing the Easysoft ODBC-Oracle Driver, [7](#)
- installing the Oracle client}, [6](#)

## L

### LibreOffice

- working with Oracle data, [41](#)

### linked server

- working with Oracle data, [39](#)

- log files, [31](#)

## M

- Materialized Views, [63](#)
- Metadata\_Dont\_Change\_Case attribute, [23](#)
- Metadata\_Dont\_Do\_Schema attribute, [24](#)
- MetaData\_ID attribute, [23](#)

## N

- No\_LOBS attribute, [25](#)
- No\_Parse attribute, [25](#)
- Node.js
  - working with Oracle data, [43](#)

## O

- OCI\_Attr\_Prefetch\_Memory attribute, [25](#)
- OCI\_Attr\_Prefetch\_Rows attribute, [25](#)
- OCI\_UTF\_Flag attribute, [26](#)
- ODBC API function support, [55](#)
- ODBC connection string attributes, [30](#)
- ODBC data sources
  - adding
    - on Linux or UNIX, [19](#)
    - on Windows, [21](#)
  - removing
    - on Windows, [18](#)
- Oracle Advanced Security, [57](#)

## P

- Password attribute, [23](#)

## Perl

- working with Oracle data, [45](#)

## PHP

- working with Oracle data, [48](#)

Pool\_Connection\_Class attribute, [28](#)

Pool\_Database attribute, [27](#)

Pool\_Increment attribute, [27](#)

Pool\_Initial attribute, [27](#)

Pool\_Max attribute, [27](#)

Pool\_Password attribute, [27](#)

Pool\_Purity attribute, [28](#)

Pool\_Scope attribute, [27](#)

Pool\_Type attribute, [27](#)

Pool\_Username attribute, [27](#)

## Power BI

- importing Oracle data, [38](#)

## PowerPivot

- importing Oracle data, [37](#)

Pull\_Lobs\_Locally attribute, [26](#)

## Python

- working with Oracle data, [50](#)

## R

## R

- working with Oracle data, [52](#)

RAC, [58](#)

## S

scalar function support, [55](#)

SQL examples, [66](#)

## SQL Server

- working with Oracle data, [39](#)

Stmt\_Caching attribute, [25](#)

## T

trace files, [31](#)

Transparent Application Failover, [59](#)

## U

### uninstalling

- on Linux or UNIX, [15](#)

- on Windows, [18](#)

Use\_Longs attribute, [24](#)

User attribute, [23](#)

## V

VarcharTrimTrailingSpaces attribute, [24](#)

## W

With\_Unicode attribute, [27](#)

## X

XA, [64](#)

XA\_Connection\_String attribute, [28](#)

XaoswName attribute, [28](#)