

Setting up SAS/ACCESS to ODBC on UNIX Platforms to Interface with ODBC-Compliant Databases

This paper describes the basic setup requirements for using the SAS System to interface with Open Database Connectivity (ODBC). It does not attempt to discuss the in-depth setup and uses of ODBC. For more information on the ODBC setup, please refer to your ODBC vendor-supplied documentation. With the assistance of this paper, you will be able to set up the interface between the SAS System and the ODBC product to gain access to a number of databases.

Brief overview of ODBC

ODBC standards provide a common interface to ODBC compliant databases. The functionality of ODBC relies on three components: the client interface, the ODBC driver manager, and the ODBC driver.

The client interface

SAS/ACCESS to ODBC software functions as the client interface in this case. SAS/ACCESS to ODBC interfaces with the ODBC driver manager, which will manage the interaction between SAS/ACCESS to ODBC and the particular ODBC driver. The following methods of accessing your data are available with SAS/ACCESS to ODBC:

- LIBNAME Access engine
- Pass-through facility

Each of these methods will be discussed in more detail later.

The ODBC driver manager (provided by a third-party ODBC vendor)

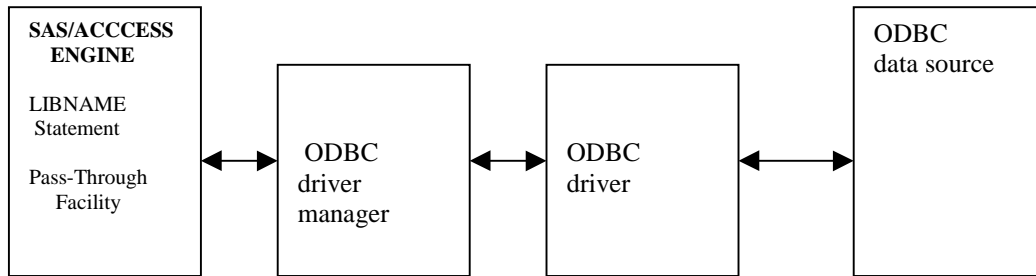
As stated above, the ODBC driver manager manages the interaction between the client interface and the ODBC driver. **SAS does not provide an ODBC driver manager, which normally ships with the specific ODBC drivers, for the UNIX platforms.** The ODBC driver manager is located in the ODBCHOME/lib directory, and for the given platforms, the names of the driver manager files are as follows:

```
Solaris.....libodbc.so
HP-UX.....libodbc.sl
AIXR.....libodbc.a
Compaq Tru64...libodbc.so
Linux.....libodbc.so
```

The ODBC driver (provided by a third-party ODBC vendor)

The ODBC drivers interface with the particular database either by directly manipulating and retrieving the data or by passing a request to a native library for a particular database. **SAS does not provide ODBC drivers for the UNIX platforms.**

The ODBC Interface to the SAS System



Basic setup to allow the SAS System to Interface with ODBC

Following is an explanation of the components of the ODBC product that must be set up and configured to allow it to interface with the SAS System. Keep in mind that this setup is also necessary for any ODBC application to successfully connect to a data source.

ODBC Data Source (dsn)

ODBC defines a data source as the data used in an application, and the operating system and network that are used to access the data. The data source contains information required to connect to a database (userid, password, driver, etc.). ODBC data sources are defined in the `odbc.ini` file. The `odbc.ini` file can contain all of your data sources, regardless of the database.

One key difference between the ODBC product on UNIX platforms as opposed to the Windows platform is that the UNIX platforms do not have an ODBC Administrator. During the UNIX ODBC installation, a generic `odbc.ini` file is created. This file can be modified to create your own unique data sources.

Following is an example of an `odbc.ini` file. It may or may not contain everything you will need in order to interface with a particular database. Please refer to your ODBC vendor-supplied documentation on the specific settings for a particular database. You may also need the documentation supplied by your database vendor as additional reference material.

Sample contents of the odbc.ini file

```
[ODBC Data Sources]
Oracle8=Oracle 8 Driver
Informix=Informix Driver
DB2=DB2 Driver
Sqlserver65=SQLServer 6 Driver
```

```
[Oracle8]
driver=/dbi/odbc/lib/ivor815.sl
Description=Oracle8
ServerName=scully_o8040
LogonID=scott
Password=tiger
```

```
[Informix]
Driver=/dbi/odbc/lib/ivinf15.sl
Description=Informix
Hostname=informixhost
LogonID=Dillon
Password=player
```

```
[DB2]
Driver=/dbi/odbc/lib/ivdb215.sl
Description=DB2
LogonID=odbc01
Password=odbc01
Location=location
Package=Default
Collection=odbc01
Protocol=TCP/IP
IPAddress=address
TcpPort=port
AddStringToCreateTable=in database odbc
MaxiumClient=10
```

```
[sqlserver65]
Driver=/dbi/odbc/lib/ivmsss15.sl
Description=MS SQL Server Version 6.5
Database=users
ServerIPAddress=10.24.35.12
ServerPortNumber=1345
LogonID=fred
Password=dino
QuotedId=Yes
AnsiNPW=Yes
```

```
[ODBC]
Trace=0
Tracefile=/tmp/odbctrace.out
TraceDll=/dbi/odbc/lib/odbctrac.sl
InstallDir=/dbi/odbc
```

UNIX environment variables

\$ODBCHOME

The value of the \$ODBCHOME environment variable is the location of the ODBC installation directory.

Korn or Bourne shell:

```
export ODBCHOME=/dbi/odbc
```

C shell:

```
setenv ODBCHOME /dbi/odbc
```

Note: There is an InstallDir parameter in the odbc.ini file. If this is defined, then the \$ODBCHOME environment variable does not need to be set. Depending on which ODBC vendor application is being used, \$ODBCHOME may not need to be defined, as some will use the value of InstallDir even if \$ODBCHOME is set. Please refer to your ODBC vendor-supplied documentation for more information, but as a rule of thumb, always set InstallDir.

```
[ODBC]
Trace=0
Tracefile=/tmp/odbctrace.out
TraceDll=/dbi/odbc/lib/odbctrac.sl
InstallDir=/dbi/odbc
```

\$ODBCINI

The value of the \$ODBCINI environment variable is the location of the .odbc.ini file. By default, the ODBC driver manager will search for the .odbc.ini (see note) file in the user's \$HOME. If the .odbc.ini file is located in the user's \$HOME, then \$ODBCINI does not need to be set. However, if \$ODBCINI is set, then the driver manager will search for the file in the specified location.

Korn or Bourne shell:

```
export ODBCINI=/dbi/odbc/.odbc.ini
```

C shell:

```
setenv ODBCINI /dbi/odbc/.odbc.ini
```

Note: The default search location is \$HOME/.odbc.ini. The file is a hidden file. If the file is named odbc.ini, the driver manager will not be able to locate it.

\$LD_LIBRARY_PATH

\$SHLIB_PATH

\$LIBPATH

These are the most important UNIX environment variables for the SAS/ACCESS to ODBC product. These are the shared library load paths, and are the paths that are searched when a specific library (in this case the ODBC driver manager and the ODBC driver) needs to be loaded. The location to be included in this UNIX environment variable is normally \$ODBCHOME/lib. In some cases, this may not be where the ODBC driver manager and drivers are installed. If this is the case, then the directory where these files reside should be entered.

For Solaris, Compaq Tru64, and Linux:

Korn or Bourne shell:

```
export LD_LIBRARY_PATH=$ODBCHOME/lib:$LD_LIBRARY_PATH
```

C shell:

```
setenv LD_LIBRARY_PATH $ODBCHOME/lib:$LD_LIBRARY_PATH
```

For HP-UX:

Korn or Bourne shell:

```
export SHLIB_PATH=$ODBCHOME/lib:$SHLIB_PATH
```

C shell:

```
setenv SHLIB_PATH $ODBCHOME/lib:$SHLIB_PATH
```

For AIXR:

Korn or Bourne shell:

```
export LIBPATH=$ODBCHOME/lib:$LIBPATH
```

C shell:

```
setenv LIBPATH $ODBCHOME/lib:$LIBPATH
```

You may also need to set database-specific UNIX environment variables. Please refer to the ODBC and/or the appropriate database vendor-supplied documentation for information on which variables should be set.

Testing the ODBC Connection

Once the ODBC environment is set up, you should test your connection outside of the SAS System to verify that it is working properly. Most ODBC vendors supply some type of query tool with their software to allow a basic connection and query of a table. If one does not exist, you should contact the ODBC vendor. Making this connection will make it easier to diagnose problems if a connection from within SAS cannot be made. Once you have verified that the connection is working outside of SAS, connectivity from SAS should be possible.

Following are some basic examples using components of the SAS/ACCESS to ODBC software. The examples use Oracle Version 8 as the data source, but the SAS syntax will be the same regardless of the data source used. For a more detailed explanation of the components of SAS/ACCESS to ODBC software, please refer to the SAS documentation.

The LIBNAME Access Engine

The following example uses the LIBNAME access engine to create an Oracle table:

```
data new;
  input name $ age;
cards;
bart 8
william 45
sue 24
megan 19
;
run;
libname test odbc uid=scott pwd=tiger dsn=oracle8;
data test.loadora;
  set new;
run;
```

This example uses the LIBNAME access engine with PROC PRINT to print the newly created Oracle table:

```
libname test odbc uid=scott pwd=tiger dsn=oracle8;
proc print data=test.loadora;
run;
```

SQL Pass-Through

The following query uses SQL Pass-through to connect to Oracle Version 8 and query the newly created Oracle table:

```
proc sql;
  connect to odbc(uid=scott pwd=tiger dsn=oracle8);
  select * from connection to odbc
    (select * from loadora);
  disconnect from odbc;
quit;
```

Interfacing SAS/ACCESS to ODBC to specific databases

The following are general setup guidelines for specific data sources. This list will be updated periodically to add other data sources as they are tested. The following examples may not contain everything that is needed to make a connection to the ODBC data source, so please refer to the vendor documentation for a more detailed description.

Redbrick

- When Redbrick is installed, the ODBC driver is not installed by default. Install the Redbrick ODBC driver from the “Install Client Connector Products” menu of the Redbrick installation.
- Redbrick does not use the \$ODBCINI UNIX environment variable. The .odbc.ini file must be placed in the user’s \$HOME directory.
- Redbrick’s rsql application **does not** use the ODBC driver.

Here is a sample UNIX environment:

```
REDBRICK_DIR=/dbi/redbrick/6.0.2.1
```

```
RB_CONFIG=/dbi/redbrick/6.0.2.1
RB_HOST=RB_THEFORCE
LD_LIBRARY_PATH=/dbi/redbrick/6.0.2.1/lib
```

Here is a sample .odbc.ini file:

```
# This section defines the ODBC environment
#
[ODBC]
InstallDir=/dbi/redbrick/6.0.2.1/lib

# This section is to name your ODBC DSNs
# One entry per DSN
#
[ODBC Data Sources]
RBDSN=Red Brick Driver

# This section is to define your ODBC DSNs
# One set of entries per DSN
#
[RBDSN]
DRIVER=/dbi/redbrick/6.0.2.1/lib/rbodbcdrv.so
SERVER=theforce.unx.sas.com:5050
RB_CONFIG=/dbi/redbrick/6.0.2.1
DATABASE=AROMA
UID=system
PWD=manager
```

Teradata

- NCR provides adhoc, an ODBC application, which can be used in testing the connection to the Teradata database.

Here is a sample UNIX environment:

```
ODBCHOME=/usr/local/odbc
ODBCINI=/usr/local/odbc/odbc.ini
LD_LIBRARY_PATH=/usr/local/odbc/drivers
```

Here is a sample .odbc.ini file:

```
[ODBC]
InstallDir=/usr/odbc
Trace=0
TraceFile=/usr/odbcusr/joe/trace.log
TraceAutoStop=1

[ODBC Data Sources]
financial=tdata.sl

[financial]
Driver=/usr/odbc/drivers/tdata.sl
```

Description=NCR 3600 running Teradata V1R5.2
DBCName=123.45.67.10
DBCName2=123.45.67.11
DBCName3=123.45.67.12
LastUser=
Username=odbcadm
Password=
Database=
DefaultDatabase=sales

Sybase

- SAS/ACCESS to ODBC may be used to access Sybase AISQ 12.x. For details, please see the following SAS Note:

<http://support.sas.com/techsup/unotes/SN/002/002435.html>
- Sybase **does not** provide an ODBC driver manager, which SAS needs, but does state in their documentation that a link can be created pointing to their driver. If you choose not to use a standard ODBC driver manager, SAS will not function with any other database via ODBC unless you first reconfigure the UNIX ODBC environment.

Here is a sample .odbc.ini file:

```
[ODBC Data Sources]
SybaseIQ=Sybase IQ ODBC driver

[SybaseIQ]
Driver=/dbi/sybase/1242/asiq12/lib/dbodbc6.so
Description=SybaseIQ
Userid=dba
Password=sql
EngineName=asiq12
CommLinks=tcip(host=10.26.10.24;port=2638)
AutoStop=no
AutoPreCommit=yes
DatabaseName=asiqusers
DatabaseFile= D:\Sybase\iq12\users.db
```

DataDirect Technologies ODBC drivers

- DataDirect Technologies provides an ODBC driver manager and ODBC drivers with support for various databases.
- DataDirect provides demoodbc, an ODBC application, which can be used in testing the connection to the specific database.

Here is a sample .odbc.ini file:

```
[ODBC Data Sources]
DB2 Wire Protocol=DataDirect 4.00 DB2 Wire Protocol Driver
SQLServer Wire Protocol=DataDirect 4.0 SQL Server Wire Protocol
Sybase Wire Protocol=DataDirect 4.0 Sybase Wire Protocol
```

[DB2 Wire Protocol]
Driver=/dbi/odbc/datadirect400/lib/ivdb217.so
Description=DataDirect 4.00 DB2 Wire Protocol Driver
LogonID=uid
Password=pwd
DB2AppCodePage=1252
ServerCharSet=1252
IpAddress=db2host
Database=db
TcpPort=50000
Package=db2package
Action=REPLACE
QueryBlockSize=8
CharSubTypeType=SYSTEM_DEFAULT
ConversationType=SINGLE_BYTE
CloseConversation=DEALLOC
UserBufferSize=32
MaximumClients=35
GrantExecute=1
GrantAuthid=PUBLIC
OEMANSI=1
DecimalDelimiter=PERIOD
DecimalPrecision=15
StringDelimiter=SINGLE_QUOTE
IsolationLevel=CURSOR_STABILITY
ResourceRelease=DEALLOCATION
DynamicSections=32
Trace=0
WithHold=0

[SQLServer Wire Protocol]
Driver=/dbi/odbc/datadirect400/lib/ivmsss17.so
Description=DataDirect 4.0 SQL Server Wire Protocol
Database=db
LogonID=uid
Password=pwd
Address=sqlserverhost,1433
QuotedId=No
AnsiNPW=No

[Sybase Wire Protocol]
Driver=/dbi/odbc/datadirect400/lib/ivase17.so
Description=DataDirect 4.0 Sybase Wire Protocol
Database=db
LogonID=uid
Password=pwd
NetworkAddress=serverhost,4100
EnableDescribeParam=1
EnableQuotedIdentifiers=0
OptimizePrepare=1
RaiseErrorPositionBehavior=0
SelectMethod=0
ApplicationUsingThreads=1

[ODBC]
Trace=0
TraceFile=odbctrace.out
TraceDll=/dbi/odbc/datadirect400/lib/odbctrac.so
InstallDir=/dbi/odbc/datadirect400
ConversionTableLocation=/dbi/odbc/datadirect400/tables
UseCursorLib=0