An overview of Open Database Connectivity (ODBC) configuration, best practices when performing these tasks, and troubleshooting techniques to assist in the resolution of common problems in this area.

Introduction

Purpose of Document

This document is designed to provide an overview of Open Database Connectivity (ODBC) configuration, best practices when performing these tasks, and troubleshooting techniques to assist in the resolution of common problems in this area.

Applicability

This document applies to IBM SPSS Modeler 15 and supported ODBC drivers installed on any platform supported by IBM SPSS Modeler Server. Many of the techniques within it can be also applied to configuration of ODBC connectivity with other IBM SPSS products, such as IBM SPSS Statistics.

Exclusions and Exceptions

The IBM SPSS Data Access Pack (SDAP) must be installed on the same host as IBM SPSS Modeler Server. In addition, if third-party ODBC drivers that are supported with IBM SPSS Modeler are being configured these must be obtained direct from the provider and installed on the IBM SPSS Modeler Server host.
Assumptions

If configuring on Linux\UNIX then familiarity with using a text editor such as vi on these systems is assumed.

Overview

Many IBM SPSS applications use ODBC to access data held in relational and other data stores. ODBC provides a standard interface for applications to access data stores without knowing the many different underlying native interfaces provided by the database management systems.

This document provides an overview of ODBC configuration, best practices when performing these tasks, and troubleshooting techniques to assist in the resolution of any problems in this functional area. This provides faster initial configuration, greater portability both between applications and version upgrades, and allows you to ensure you have the correct resources and information are available to successfully complete this stage in your application setup.

This document does not go into any detailed ODBC driver configuration or cover configuration options. This information can be obtained from the ODBC driver documentation if required.

ODBC driver support with IBM SPSS applications

Stability in the ODBC interface is critical for the successful functioning of many IBM SPSS applications that access data stored within databases. A comprehensive list of ODBC drivers and databases that are supported by IBM SPSS products can be found in the Resources section at the end of this document.

ODBC drivers – what is supported

The IBM SPSS Data Access Pack

The IBM SPSS Data Access Pack (SDAP) provides ODBC drivers from DataDirect that are for exclusive use with IBM SPSS applications. The supported version of SDAP with each product version is available for download as part of the product E-Assembly. There are SDAP packages for each server platform that is supported, and if both 32 and 64-bit versions of an application is available on a platform, there are 32-bit and 64-bit versions of the SDAP for that platform. 32-bit ODBC drivers do not work with 64-bit applications and vice-versa.

NOTE: There are ODBC drivers contained within the SDAP that are not supported with SPSS products. A link to the list of the drivers and databases that are supported by IBM SPSS products can be found in the Resources section at the end of this document.

Non-SDAP provided ODBC drivers

In addition to the SDAP-delivered drivers, there are several ODBC drivers provided by database vendors that are supported by IBM SPSS applications. These usually, but not always, cover those databases for which there is no SDAP equivalent driver. IBM does not provide the non-SDAP
ODBC drivers to IBM customers - it is the customer's responsibility to obtain the supported driver version from the database vendor directly. You must still install the SDAP even if using non-SDAP provided ODBC drivers as the SDAP provides the driver manager.

**ODBC driver categories**

There are two broad categories of ODBC drivers.

- **Wire Protocol** – self contained (no external dependencies), easier to install and configure.
- **Thick-client** - These drivers have external dependencies. For the purposes of this document any ODBC driver without the words “Wire Protocol” in the name are referred to as thick-client drivers.

The only requirement on the application server to use a Wire Protocol driver is the SDAP itself. The thick-client drivers on the other hand have additional dependencies on other database client software. This database client software must be installed and available on the application server along with the ODBC driver in order to be able to use ODBC. All of the SDAP supported drivers are Wire Protocol with the exception of the SDAP driver for Teradata. Database client software requirements for the supported thick-client drivers can be found in the driver vendors' documentation for the non-SDAP drivers. For example, the link to DataDirect's website or the SDAP Teradata drivers can be found in the Resources section at the end of this article.

**Installing ODBC Drivers**

**Installing the SDAP drivers**

To install the SDAP ODBC drivers extract the contents of the downloaded SDAP image that came with your product downloaded files and follow the installation instructions within that package that apply to the relevant platform for your application server host.

Running the SDAP installer on Linux\Windows only makes the gzipped .TAR file available on the host machine – it does not perform the full install of the ODBC drivers. Follow the installation instructions within the SDAP package for details of how to extract the drivers.

On non-Windows environments it is imperative that you run the `setodbcpath.sh` script as documented in the SDAP installation instructions after extracting the drivers. Failure to do this can have various adverse effects further along in your configuration or use of the software.

**Installing driver dependencies and any non-SDAP ODBC drivers**

Any database client software required by the driver must also be installed and configured on the application server host. If installing on a Windows platform, a reboot of the server is usually required after the installation of the database client software to make it available to applications. On UNIX\Linux platforms, you must ensure that the ODBC environment within the application is aware of any database client requirements. See the section titled Defining the ODBC environment below for further details. If you are using non-SDAP supplied ODBC drivers, these need to be obtained from the database vendor and installed, in addition to the SDAP.
Configuring ODBC drivers

Configuring an ODBC DSN

Once installed, the ODBC drivers must be configured with the information about the underlying datasource(s). This is done using an ODBC Data Source Name (DSN). Within the DSN you define various parameters required to connect to the database. The DSN is the resource used by the application to connect to the datasource defined within it.

Where to define the ODBC DSN's

For Window environments, DSNs are defined in the Microsoft ODBC Datasource Administrator. A GUI is provided by the drivers that can be used to specify the connection parameters.

For UNIX\Linux environments, DSNs are defined in a text file usually named odbc.ini. There is a default odbc.ini installed with the SDAP. ODBC DSN's are defined in this file using a text editor such as vi. On UNIX\Linux platforms it is good practice to create a backup of the files odbc.ini and odbc.sh immediately after installation and initialisation and before editing them to define your ODBC environment. This allows you to return to the default files if the need arises.

ODBC configuration information for your datasource(s) should be obtained from your database administrator (DBA).

The base parameters required for each supported driver are detailed in Table 1. The Windows parameter label is listed followed by corresponding parameter names in UNIX odbc.ini in square brackets. This list should not be used as a reference for supported drivers as support can change from release to release. This is a reference of most frequently used drivers.

Table 1: Mandatory connection parameters for each driver

<table>
<thead>
<tr>
<th>ODBC Driver</th>
<th>Required connection parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB2 Wire Protocol</td>
<td>Ip Address [IpAddress]</td>
</tr>
<tr>
<td></td>
<td>Top Port [TcpPort]</td>
</tr>
<tr>
<td></td>
<td>Database Name [Database]</td>
</tr>
<tr>
<td></td>
<td>Location Name [Location]</td>
</tr>
<tr>
<td></td>
<td>Collection [Collection]</td>
</tr>
<tr>
<td>IBM DB2 ODBC Driver</td>
<td>Database [Database]</td>
</tr>
<tr>
<td></td>
<td>Protocol [Protocol]</td>
</tr>
<tr>
<td></td>
<td>Port [Port]</td>
</tr>
<tr>
<td></td>
<td>Hostname [Hostname]</td>
</tr>
<tr>
<td>Teradata</td>
<td>DBCName or Alias (SDAP Teradata driver) [DBCName]</td>
</tr>
<tr>
<td></td>
<td>Name(s) or IP Addresses (Teradata's driver) [DBCName]</td>
</tr>
<tr>
<td>SQL Server Wire Protocol</td>
<td>Server [Address]</td>
</tr>
<tr>
<td></td>
<td>Database Name [Database]</td>
</tr>
<tr>
<td>SQL Server Native Client</td>
<td>Server [not available on non-Windows]</td>
</tr>
<tr>
<td>Sybase IQ</td>
<td>Server Name [Servername]</td>
</tr>
</tbody>
</table>

Further details of configuring Modeler to use this Driver on UNIX\Linux can be found at [http://www.ibm.com/support/docview.wss?uid=swg21618950](http://www.ibm.com/support/docview.wss?uid=swg21618950)
In addition to the mandatory connection parameters above, Table 2 below details several driver specific options that are required or optional to deal with specific use scenarios. On Windows you have to navigate away from the main tab when configuring the ODBC DSN to enable these options. Windows parameter label is listed followed by corresponding parameter names in UNIX odbc.ini in square brackets.

Table 2: Driver specific settings based on use scenarios

<table>
<thead>
<tr>
<th>ODBC Driver</th>
<th>Required/optional parameters</th>
<th>When to enable the option and why</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL Server Wire Protocol</td>
<td>&quot;Enable Quoted Identifiers&quot; checked (Advanced tab) [EnableQuotedIdentifiers=1]</td>
<td>If the application uses quoted identifiers – IBM SPSS Modeler does this by default. Errors will occur on query execution without this option enabled.</td>
</tr>
<tr>
<td>Informix Wire Protocol</td>
<td>&quot;Use Delimited Identifier&quot; – checked (Advanced tab) [UseDelimitedIdentifiers=1]</td>
<td>If you are creating tables in Informix from IBM SPSS Modeler where the column names are not comprised of only lowercase characters. Errors will occur on query execution without this option enabled.</td>
</tr>
<tr>
<td>DB2 Wire Protocol</td>
<td>&quot;Use Current Schema For Catalog Functions&quot; – checked (Advanced tab) [UseCurrentSchema=1]</td>
<td>If you have a very large number of objects returned in the select table/view dialog and you know that all the objects you require are within the default schema of the user you are using to connect to DB2. Performance of populating this dialog should improve with this option enabled with less objects being returned.</td>
</tr>
<tr>
<td>Teradata</td>
<td>&quot;Session Mode&quot; = ANSI (&quot;options&quot; button exposes this) [DriverUnicodeType=1]</td>
<td>If you are running Teradata in &quot;Teradata&quot; mode that does not allow duplicate values within a column. This overrides the connection so that duplicates can be inserted into tables. Inserts that include duplicate values within a column will fail without this override when Teradata is not running in ANSI mode.</td>
</tr>
</tbody>
</table>

Further details of the parameters within the DSN can be found in the DataDirect Connect Series for ODBC User’s Guide (installed with the SDAP) for the SDAP ODBC drivers, and in vendor-
specific ODBC driver documentation for the supported non-SDAP drivers. Please contact the
database vendor for any vendor-specific ODBC configuration documentation.

Defining the ODBC environment

In addition to defining the relevant connection attributes within an ODBC DSN, the ODBC
environment needs to be set so that the driver can be used to make a connection to the
datasource. Failing to do this correctly will render the drivers unusable. This is simpler on Windows
than on non-Windows platforms and is simpler with the Wire Protocol drivers than with the thick-
client drivers. At a very high level, setting the ODBC environment involves:

- Creating a link to the relevant driver manager from within Modeler Server on a UNIX\Linux
  platform. For details see the Configuring ODBC drivers on UNIX systems section of the
document IBM SPSS Modeler Server 15 for UNIX Installation Instructions.
- Ensuring that the application using the ODBC DSN's that you have defined knows where
to look for them. On UNIX\Linux the application will look for DSNs in the file defined by the
  ODBCINI environment variable which, by default, is set and exported in the odbc.sh script.
- For thick-client drivers only, ensuring any additional dependencies are on the system PATH
  on Windows platforms, or the location of them are specified on UNIX\Linux environments.
  Environment variables are used to do this.

Post installation steps

After configuring the drivers there may be additional post-installation steps required. Table 3 for
Windows installations and Table 4 for Unix\Linux list out possible post-install steps.

Table 3: Windows installation post-install steps

<table>
<thead>
<tr>
<th>Type of Driver</th>
<th>Post install steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wire Protocol Driver</td>
<td>No further action required</td>
</tr>
<tr>
<td>Thick-Client Driver</td>
<td>Driver dependencies must be installed - system reboot often necessary after this.</td>
</tr>
</tbody>
</table>

Table 4: Unix\Linux installation post-install steps

<table>
<thead>
<tr>
<th>Type of Driver</th>
<th>Post install steps</th>
</tr>
</thead>
</table>
| Wire Protocol Driver | 1. ODBC environment will be defined within the odbc.sh script installed with the drivers.  
2. Sourcing this script from any application enables the application to use the drivers. No additional editing of odbc.sh should be required.  
3. SPSS Modeler Server 15 on non-Windows platforms requires you to link a wrapper to the relevant ODBC driver manager before using ODBC. |
| Thick-Client Driver  | 1. Driver dependencies must be installed on the system. ODBC environment will be defined within the odbc.sh script installed with the SDAP drivers.  
2. The odbc.sh script should be edited to append the location of any driver dependencies to the library path environment variable. Sourcing this script from any application enables the application to use the drivers.  
3. Driver-specific environment settings may be required. For example, Netezza requires NZ_ODBC_INI_PATH to be exported.  
4. Link appropriate libspssodbc_datadirect* wrapper to the ODBC driver manager as documented in the section Configuring ODBC drivers on UNIX systems of the document IBM SPSS Modeler Server 15 for UNIX Installation Instructions. If using SAP Hana or the IBM DB2 ODBC Driver please refer to SPSS Modeler 15.0 FP1 release notes for further information. |
Testing ODBC datasources outside of the application

Once you have configured an ODBC datasource and setup the ODBC environment, it is good practice to check the configuration outside of any applications that you intend to use the DSN from.

For Windows, this is normally done using a test connect button provided within the Microsoft ODBC Datasource Administrator. Use this button to determine that a connection can be made to the datasource with an ODBC DSN that you have configured before trying to use the DSN from within an IBM SPSS application. If errors are returned by the test connect, then work with your DBA to resolve the issue.

For UNIX\Linux, a test program is provided with the SDAP drivers that can be used to connect to the datasource via a defined ODBC DSN outside of IBM SPSS applications. Here are the steps to use this test program:

1. Navigate to the SDAP installation directory.
2. Execute the odbc.sh script with the command `./odbc.sh`.
3. Issue the command `echo $ODBCINI` to ensure values are what was set in odbc.sh.
4. Navigate to the `./samples/example` subdirectory.
5. Execute the example command line program with the command `./example`.
6. Specify the ODBC DSN and the login of the database when requested.

If the connection is successful then an SQL> prompt will appear after the example program makes a connection to the datasource. If the connection fails then it indicates a problem within the ODBC environment defined in odbc.sh and odbc.ini files. Take steps to rectify the connection failure before moving on to using the ODBC DSN from within the application.

Additional tips when testing and configuring ODBC drivers:

- With the Wire Protocol ODBC drivers on non-Windows systems, no editing of odbc.sh should be necessary in order to setup the required ODBC environment. If using thick-client ODBC drivers it is likely that odbc.sh will need to be edited as part of the configuration process to detail any driver dependencies. The location of these should be appended to the definition of the library path variable for the platform in question within odbc.sh. Appendix A details a specific example of doing this.
- On Windows 64-bit systems there are 32-bit and 64-bit Microsoft ODBC Datasource Administrator applications that are used to configure 32-bit and 64-bit ODBC DSNs respectively. Please ensure that you install the correct ODBC drivers for the application you intend to use them with and configure them with the correct Microsoft ODBC Datasource Administrator. If you do not see the ODBC DSN's you expect from within a Windows application installed on Windows 64-bit it is usually because the 32-bit drivers have been installed and configured and you are using a 64-bit application.
- On Windows systems there are the concepts of System and User DSN's. If you are running a server product that runs as a Windows service you need to configure an ODBC DSN as a System DSN in order for it to be available to all users of that server. User DSN's are visible and available only to the user that created them.
Configuring applications to use the ODBC drivers

Once the test connections to the database are successful outside of the IBM SPSS application, the application itself needs to have its environment configured to be able to use the same ODBC driver.

For Windows, ODBC compliant applications will automatically pick up the defined ODBC DSNs by virtue of their definition in the system registry.

For UNIX/Linux, ODBC compliant applications need to be made ODBC aware of any defined ODBC DSNs by manual configuration of the ODBC environment. This is done by setting and exporting environment variables within the application environment. It is best practice to do this within separate shell scripts from the main application startup script. These ODBC environment scripts can then be sourced from the application startup script. This makes the ODBC environment definition portable so it can be sourced from multiple locations and also means you do not risk introducing other errors in the main application server startup script whilst trying to configure the ODBC environment. The SDAP installs a script named odbc.sh that is generally used for this purpose.

The ODBC environment script on UNIX/Linux

It is recommended to append the path to any required dependencies of a non-Wire Protocol ODBC driver to the system library path definition within odbc.sh. This is a safer and more efficient way to configure the environment than appending these additional paths within the application startup script after sourcing the default odbc.sh.

Sourcing the ODBC environment script from the application startup script on UNIX/Linux

Edit the modelersrv.sh script in the SPSS Modeler Server installation directory and add a line to source the odbc.sh script which was edited prior. It is recommended to make this change immediately below the close of the if loop that sets the ODBC_DRIVER_MANAGER_PATH environment variable as shown in Figure 1 below.

Figure 1: Sourcing an odbc.sh script

```
if [ -z "$(ODBC_DRIVER_MANAGER_PATH)" ]; then
    DLLIBPATH=$(ODBC_DRIVER_MANAGER_PATH):DULLIBPATH; export DLLIBPATH
fi

./space/support_sh/sdap_mod15/odbc.sh
```

The syntax used is essential and a full path to the odbc.sh file must be preceded with a period and a space as such:

```
./space/support_sh/sdap_mod15/odbc.sh
```
The application must be restarted after making edits to the startup script in order for them to take effect.

**Testing the connection from within the application**

From IBM SPSS Modeler Client, connect to the SPSS Modeler Server that you have just configured to use ODBC. Invoke the database connection dialog from the Tools > Databases menu. You should see the DSN that you have defined in the Data Sources list. Select the DSN, specify the database login in the username and password fields and click Connect. If the connection is successful it will appear in the Connections panel.

**Troubleshooting connection issues and errors**

Below are some common errors and resolutions when configuring ODBC.

**Cannot open shared object file libodbc.so, preventing SPSS Modeler Server startup**

After invoking the modersrv.sh script, the SPSS Modeler Server is failing to start after defining the ODBC environment and there are references to `libodbc.so` in the error message. For example,

```
[root@MMRH 15.0] # ./modelersrv.sh start
/usr/IBM/SPSS/ModelerServer/15.0/modelersrv_15_0: error while loading shared libraries: libodbc.so: cannot open shared object file: No such file or directory
IBM SPSS Modeler Server failed to start
[root@MMRH 15.0] #
```

This error indicates that the load library path environment variable for the OS (which is appended to with the relevant SDAP paths by the odbc.sh) is incorrectly set and does not include the path to the SDAP/lib subdirectory that is required.

Verify that the odbc.sh script is correctly sourced from within your SPSS Modeler Server startup script and that you have restarted SPSS Modeler Server.

**Empty datasources list in Database Connections dialog when connected to SPSS Modeler Server**

No entries are listed in the Data sources: list in the Database Connections dialog box as shown in Figure 2 below.

**Figure 2: Database Connection window in SPSS Modeler Server**
**Windows:**

- The DSN entry created is a System DSN and not a User DSN.
- A 64-bit DSN for SPSS Modeler Server 64-bit and/or a 32-bit DSN for SPSS Modeler Server 32-bit was created.

On 64-bit Windows systems there are separate 32-bit and 64-bit Microsoft ODBC Datasource Administrator tools. The 32-bit tool will allow configuration of any 32-bit ODBC drivers installed on the system, the 64-bit tool is used for the 64-bit ODBC drivers. To invoke the 32-bit tool run **C:\WINDOWS\SysWOW64\odbcad32.exe**. Any drivers and DSNs you see displayed here are 32-bit DSNs and will not be available to 64-bit SPSS Modeler Server or Client. The 64-bit tool can be invoked by running **C:\WINDOWS\system32\odbcad32.exe** in order to verify you have the expected 64-bit drivers installed and DSNs configured.

**UNIX\Linux:**

- The odbc.sh script was sourced properly from within the SPSS Modeler Server startup script using the correct syntax.
- The SPSS Modeler Server has been restarted after making changes.
- The necessary links to the relevant ODBC wrapper after installing SPSS Modeler Server have been created.
- It is possible to connect to the same DSN using the test program outside of SPSS Modeler.

**Unexpected DSN’s listed in datasources list in Database Connections dialog when connected to SPSS Modeler Server**

Verify the following depending on your platform.

**Windows:**

- Check the points listed in the previous error.

**UNIX\Linux:**

- You have defined your ODBC DSN within the file specified by the **ODBCINI** environment variable in the odbc.sh script. If you have defined it elsewhere you must either append the DSN to the file specified by ODBCINI in the odbc.sh script (recommended) or edit the odbc.sh script to re-specify ODBCINI to point to the file in which you have defined your ODBC DSN(s). Restarting SPSS Modeler Server is necessary after making these changes.

**Data Source name not found and no default driver specified**

This error is a result of mismatched datasource names on UNIX\Linux systems.

When defining a DSN in a UNIX\Linux odbc.ini, the datasource name specified in the [ODBC Data Sources] section must match exactly the datasource name specified within square brackets above the section in which you are defining the parameters for the DSN. If there is any mismatch
between these you will see this error in when attempting to make a connection to the datasource. The following is an example of how both names use ‘DB2 Wire Protocol’ exactly.

```
[ODBC Data Sources]
DB2 Wire Protocol=IBM Corp. 6.1 DB2 Wire Protocol

[DB2 Wire Protocol]
Driver=/usr/IBM/SPSS/SDAP/lib/XEdb225.so
Description=IBM Corp. 6.1 DB2 Wire Protocol
AccountingInfo=
AddStringToCreateTable=
```

Eliminate any such mismatches from odbc.ini and ensure the example program can be used to successfully connect to the datasource as detailed earlier in the section titled **Testing ODBC Datasources outside of the application** before trying again from within SPSS Modeler Server.

**Specified driver could not be loaded**

This error is often a result of an incorrect value for the **Driver=** parameter in the DSN.

A common cause of this error is when the setodbcpath.sh script has not been executed after installing the SDAP as detailed earlier. If this step is missed the **Driver=** parameters within all the default DSN’s in the odbc.ini will contain references to the hard-coded path ODBCHOME, which does not exist.

Verify that the driver parameter points to a valid file on the system and ensure the example program can be used to successfully connect to the datasource before trying again from within SPSS Modeler Server.

There are additional causes of this error, particularly when using the thick-client ODBC drivers. Refer to additional diagnostic steps in the example error “Connection failures with non-Wire Protocol ODBC drivers -specified driver could not be loaded.”

**ServerName cannot be used with HostName, PortNumber, SID, or ServiceName**

This error is the result of incorrectly defining multiple, mutually exclusive parameters for an Oracle Wire Protocol DSN on UNIX/Linux system.

Verify that the connection parameters specified in the error are not all specified within the same DSN. The default odbc.ini that ships with the SDAP contains comments within inequalities for some parameters to assist in DSN configuration. Any unused comments must be removed from the DSN definition within odbc.ini or the corresponding parameters must be commented out of the file.

To comment out a parameter use the hash character at the start of the relevant line. For example, 
```
#TNSNamesFile=<tnsnames.ora_filename>
```

To remove the comment value from a parameter definition, delete the string including the inequalities, so that the parameter is set to an empty value. For example 
```
TNSNamesFile=
```
With the Oracle Wire Protocol Driver you can connect either via TNS (if you have Oracle client installed and configured) or directly to the server specifying HostName, PortNumber and the SID of the instance or the ServiceName (global database name).

On Windows platforms you are not able to misconfigure a DSN in this way as the mutually exclusive slots are disabled in the GUI when others are populated.

**Connection refused**

This error is often a result of incorrectly defined Oracle Wire Protocol DSN parameter values. “Connection refused” errors are usually followed by error text that should be reviewed with the database administrator (DBA) in combination with the DSN definition. For example, the error “Verify Host Name and Port Number” states that a connection to the Oracle database specified in the DSN cannot be made and reason for that is an invalid combination of values of the HostName and PortNumber parameters.

**Unknown error: SQLCODE -1001**

This error is a result incorrectly defining multiple mutually-exclusive parameters for a DB2 Wire Protocol DSN on a UNIX\Linux system. Verify that you are specifying only the DSN parameters relating to the DB2 platform that you are attempting to connect to (see Table 1).

The default odbc.ini that ships with the SDAP contains comments within inequalities for some parameters to assist in DSN configuration. Any unused comments must be removed from the DSN definition within odbc.ini or the corresponding parameters must be commented out of the file. For example, if you are attempting to connect to DB2 on UNIX\Linux or Windows and have not removed the **Location** and **Collection** parameters from the DSN definition this error will result.

On Windows platforms you are not able to misconfigure a DSN in this way as the mutually exclusive slots are disabled in the GUI when others are populated.

**Errors during connection or execution that return error codes rather than descriptive text strings on UNIX\Linux**

Errors or warnings that have a number instead of a descriptive message are indicative of configuration issues in the ODBC environment. This is caused by the drivers being unable to find their message files and having no way to map the error codes to error text. The **InstallDir** parameter in the [ODBC] section of the odbc.ini is used by the drivers to map codes to text. In a correctly installed ODBC environment this parameter will be set to the installation directory of the SDAP. If the setodbcpath.sh script is not executed after installing the SDAP as described earlier, this parameter will be incorrectly set for the ODBC installation. Please verify that this parameter is correctly set to your SDAP installation directory.

**ODBC test program on UNIX\Linux connecting but SPSS Modeler Server connection attempt to same DSN hanging**

You have configured a DSN using an SDAP driver and can use it to successfully connect to the datasource using the ODBC test program as detailed earlier. When an attempt to make a
connection using the same DSN from within SPSS Modeler Server with the identical environment configured, the connection attempt hangs.

Verify the user profile used in testing the ODBC connection is the same profile currently being used to connect to the server. If it is not, compare the two profiles with special attention to the authorities of each.

Verify the `ODBC_DRIVER_MANAGER_PATH` environment variable in the SPSS Modeler Server startup script is not set. This environment variable should no longer be set in the Modeler Server startup script, contrary to the comments in the script, regardless of which Teradata ODBC driver you are using. If you have set this to a path on the system, back-out the change and then restart SPSS Modeler Server for it to take effect.

**Connection failures with non-Wire Protocol ODBC drivers - specified driver could not be loaded.**

The thick-client ODBC drivers (Teradata driver from the SDAP or the supported non-SDAP drivers) have additional dependencies that must be installed and configured on the ODBC client as detailed earlier. If this is not done and you attempt to make a connection with the drivers you will likely see the errors such as “Server and/or port attributes are empty” or “Specified driver could not be loaded”.

Please verify that the documented ODBC driver dependencies have been installed onto the ODBC client machine and for a Windows environment ensure the server machine has been rebooted and for a UNIX\Linux environment ensure the path to these dependencies have been appended to the library path variable specification in the odbc.sh script and SPSS Modeler Server has been restarted.

A detailed walk-through of this process with the SDAP Teradata driver on Linux is provided in Appendix A.

**Connection failures with non-Wire Protocol drivers – missing environmental specification on UNIX\Linux**

With some non-Wire Protocol drivers, when the ODBC driver cannot find the definition of the DSN that has been selected it will return a message stating attributes are empty. For example, the message might be “Server and/or port attributes are empty. Failed to connect to datasource: Netezza”.

If using Netezza, verify that you have set and exported `NZ_ODBC_INI_PATH` to the location of the directory in which your odbc.ini resides within the odbc.sh script and also that you have copied the Netezza driver definition from the odbcinst.ini.sample file installed with the Netezza driver into the SDAP odbcinst.ini. This requirement is documented in the Netezza ODBC driver `ODBC_README` file and illustrates the need to follow any driver-specific documentation during configuration.
If using the Netezza DSN exactly as copied from the Netezza sample file you must remove the space between the default name and the equal sign in the ODBC Data Sources stanza used by Modeler as shown below:

```
[ODBC Data Sources]
NZSQL = NetezzaSQL
Change to:
[ODBC Data Sources]
NZSQL= NetezzaSQL
```

### Appendix A – Configuration of driver dependencies on UNIX\Linux

This process is with specific reference to the SDAP Teradata driver on Linux but the steps detailed can be applied to troubleshoot connection issues with any thick-client ODBC driver.

**Scenario**

Attempts to connect from SPSS Modeler client using the SDAP Teradata driver fail with the message “Specified driver cannot be loaded”.

**Steps to fix the issue**

1. Confirm the same error is returned by the ODBC example tool. From SDAP install execute odbc.sh from within the SDAP installation directory, navigate to the samples/example directory and execute the example binary as shown below.

   ```
   #. ./odbc.sh
   #cd samples/example
   #./example
   ./example DataDirect Technologies, Inc. ODBC Example Application.
   Enter the data source name : Teradata
   Enter the user name        : mh
   Enter the password         : xxxxx
   SQLSTATE = IM003
   NATIVE ERROR = 0
   MSG = [DataDirect][ODBC lib] Specified driver could not be loaded
   ```

   In this case the error is the same.

2. Use the ddtestlib utility to try to load the driver library from a shell.
   - Copy the value of the Driver parameter within the DSN specification you are attempting to use from within your odbc.ini. In this scenario, this value will be `/usr/IBM/SPSS/SDAP/lib/XEtera25.so`.
   - From the same shell (as this is where the environment defined in odbc.sh that you executed in step 2 has been setup), navigate to the SDAP `/bin/bin` subdirectory and execute the ddtestlib binary with the driver library as an argument. In this scenario, the command line would be `./ddtestlib /usr/IBM/SPSS/SDAP/lib/XEtera25.so` and would result in the following error - Load of /usr/IBM/SPSS/SDAP/lib/XEtera25.so failed: libtdgss.so: cannot open shared object file: No such file or directory. The error is stating that a Teradata library required by the SDAP Teradata driver, `libtdgss.so`, cannot be found.

3. Update the library path in the odbc.sh script so the library can be found.
• Find the location of the library referenced in the error and append the path to the library path that is setup within the odbc.sh script using the “:” path separator. This is also shown as an example in Figure 3 below.

**Figure 3 - Appending path to the odbc.sh script**

If you are unable to find the library on the system then there are still driver dependencies that are not installed on the system. Review the driver documentation and correct this.

• Save and re-execute odbc.sh from within the shell. Once done issue the command `echo $LD_LIBRARY_PATH` to ensure your change has been applied. In this scenario, the updated path would read,

```
/usr/IBM/SPSS/SDAP/lib:/usr/IBM/SPSS/SDAP/lib:/usr/IBM/SPSS/SDAP/lib:/usr/IBM/SPSS/SDAP/lib:/opt/teradata/teragss/suselinux-x8664/13.10.04.01/lib/
```

• Rerun ddtestlib as detailed in step 2. The resulting message should appear similar to,

```
Load of /usr/IBM/SPSS/SDAP/lib/XEtera25.so successful, qehandle is 0x887040
File version: 06.12.0073 (b0066, U0066)
```

Now the driver library has been loaded successfully so you can proceed with trying again to connect with the ODBC example tool as detailed in step 1. If ddtestlib shows failure to load other libraries, repeat the steps in steps 2 and 3 until it shows successful loading of the library before proceeding.

4. Restart SPSS Modeler Server and attempt a connection using the DSN.

• Once the ODBC example tool can successfully connect with the environment specified in the odbc.sh, then restart SPSS Modeler Server and you should now be able to connect. The SPSS Modeler Server restart is mandatory for the edits to odbc.sh that you have made to be propagated into the SPSS Modeler Server environment.
Resources

- List of IBM SPSS products and supported drivers and databases
- IBM SPSS Modeler 15.0 FP1 release notes
- SDAP Teradata software requirements from DataDirect
About the author

Matt Harvey

Matt Harvey has been a solution consultant in SPSS technical support for 12 years, working primarily with SPSS Modeler. Throughout this time he has specialised in enabling customers with implementations and integrations between various products, both SPSS and third-party.

© Copyright IBM Corporation 2013, 2014
Trademark
(www.ibm.com/developerworks/ibm/trademarks/)