Get Connected - Attach To Your Data Sources Using SAS/ACCESS

Laura Liotus, Community Care Behavioral Health, Pittsburgh, PA

ABSTRACT

In today's computing environment, having to connect to data sources besides SAS ® data sets is common. Community Care Behavioral Health SAS users have the need to connect to a multitude of data sources. These sources include SAS data sets and Microsoft products including SQL Server, Access and Excel. We also have Oracle databases and flat files. SAS provides many SAS/ACCESS engines to attach to these sources. For some of the sources, there are multiple engines to choose from. In addition to multiple engines, there are various syntactical ways to attach to the data sources using an engine. In researching the various ways to connect to the data sources at Community Care, I had to contact SAS Support, search the SAS Support Knowledge Base and various other online resources.

This paper compiles and demonstrates some of the ways we connect to data sources at Community Care using the SAS/ACCESS engines for which we are licensed.

INITIAL SAS STEP TO GET CONNECTED

The first step is to check which SAS/ACCESS licenses your company has acquired. This can be done by running the following in Base SAS.

```
PROC SETINIT; RUN;
```

```
---SAS/STAT                                          31OCT2010
---SAS/GRAPH                                         31OCT2010
---SAS/IML                                           31OCT2010
---SAS/SHARE                                         31OCT2010
---SAS/CONNECT                                       31OCT2010
---SAS/SHARE*NET                                     31OCT2010
---SAS Integration Technologies                      31OCT2010
---SAS Data Quality Server                           31OCT2010
---SAS/ACCESS Interface to ORACLE                    31OCT2010
---SAS/ACCESS Interface to ODBC                      31OCT2010
---SAS/ACCESS Interface to OLE DB                    31OCT2010
---SAS Data Quality Server - English (United States) 31OCT2010
---SAS Metadata Bridges for General Industry Standards 31OCT2010
---SAS Stat Studio                                   31OCT2010
---SAS Workspace Server for Local Access             31OCT2010
---SAS Workspace Server for Enterprise Access        31OCT2010
```

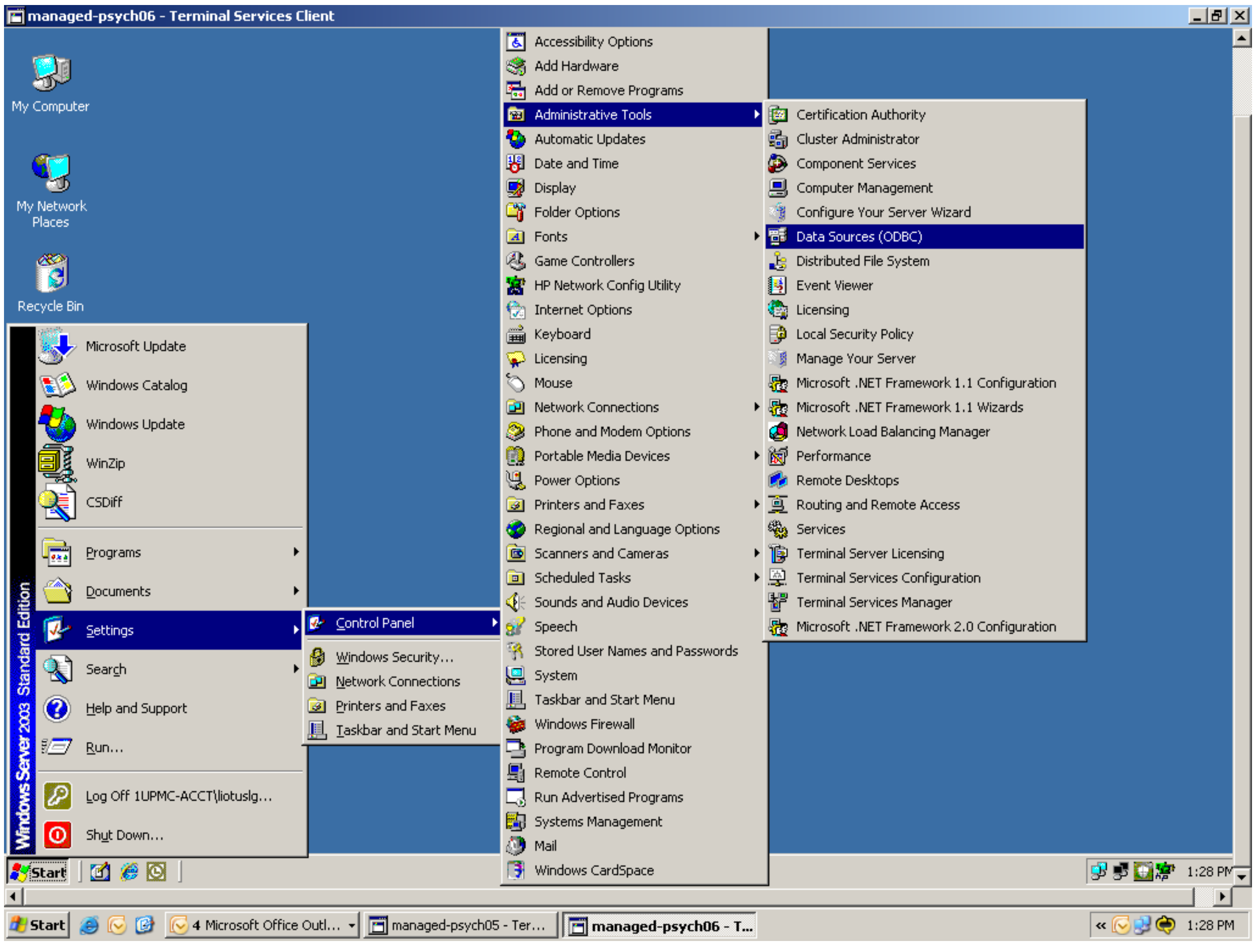
As the output indicates, we have SAS/ACCESS Interface to ORACLE, SAS/ACCESS Interface to ODBC and SAS/ACCESS Interface to OLE DB.

I will focus on ODBC, OLE DB and SQL Pass-through using the SAS/ACCESS Interface to ORACLE.

GET CONNECTED USING GUI

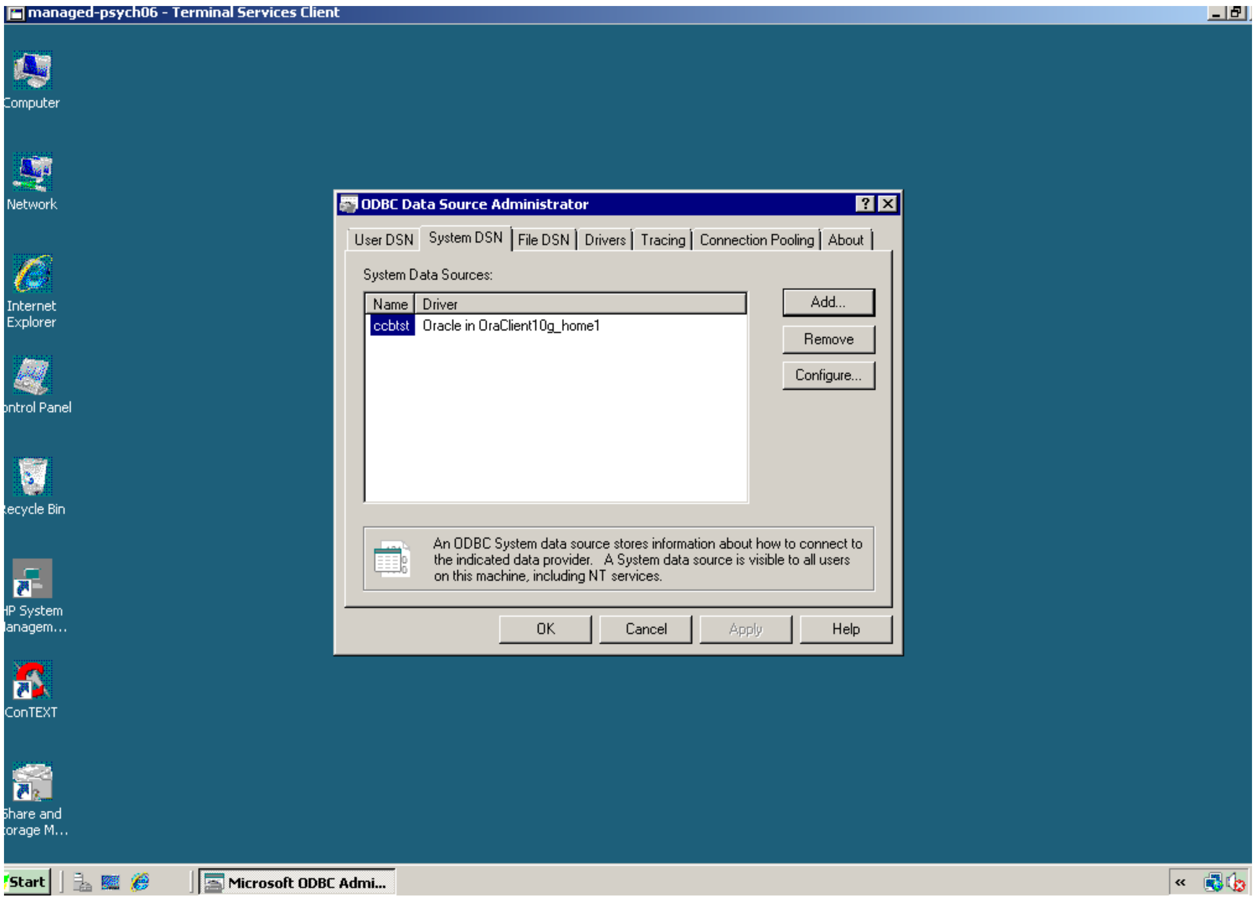
The first order of business is setting up the ODBC connection. This example ODBC connection is for Oracle. Whether connecting to Microsoft SQL Server, Access or Excel, the ODBC set-up is very similar. Usually login and password are not needed for Access and Excel.

I am amazed how many IT folks don't know how to set-up an ODBC connection. So, I apologize if this is too elementary but just in case you don't know how to configure this connection, what follows is an example.



Step 1: The ODBC configuration is included with the Windows operating system. For Oracle, unless you want to use the Microsoft ODBC driver for Oracle, you will have to install the Oracle client. If you don't have access to the software, have whoever is responsible for such a task in your environment install the client that will include the Oracle ODBC driver.

Click through the Start Menu/Settings/Control Panel/Administrative Tools/Data Sources (ODBC) to begin the configuration.

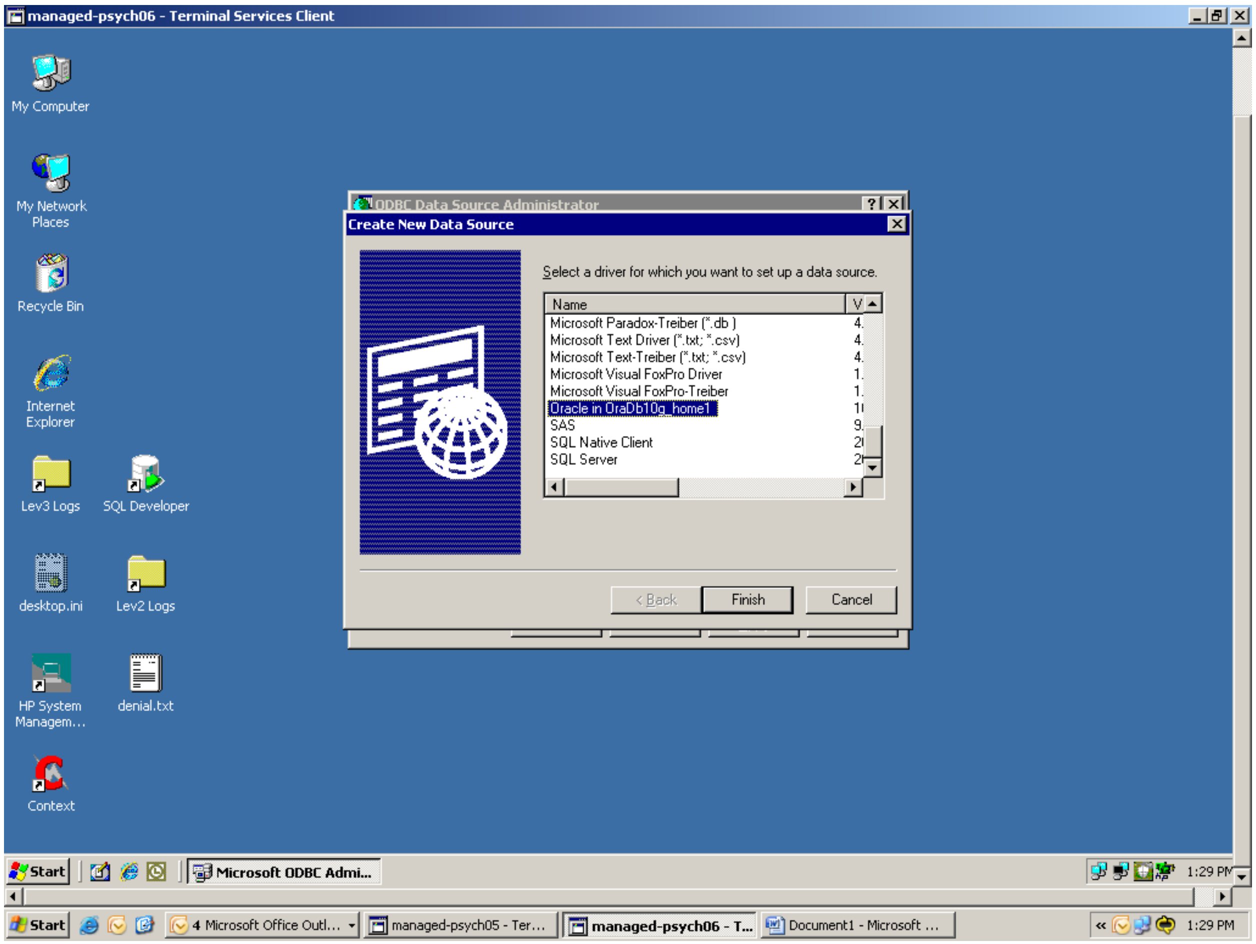


System Data Sources:

Name	Driver
ccbtst	Oracle in OraClient10g_home1

An ODBC System data source stores information about how to connect to the indicated data provider. A System data source is visible to all users on this machine, including NT services.

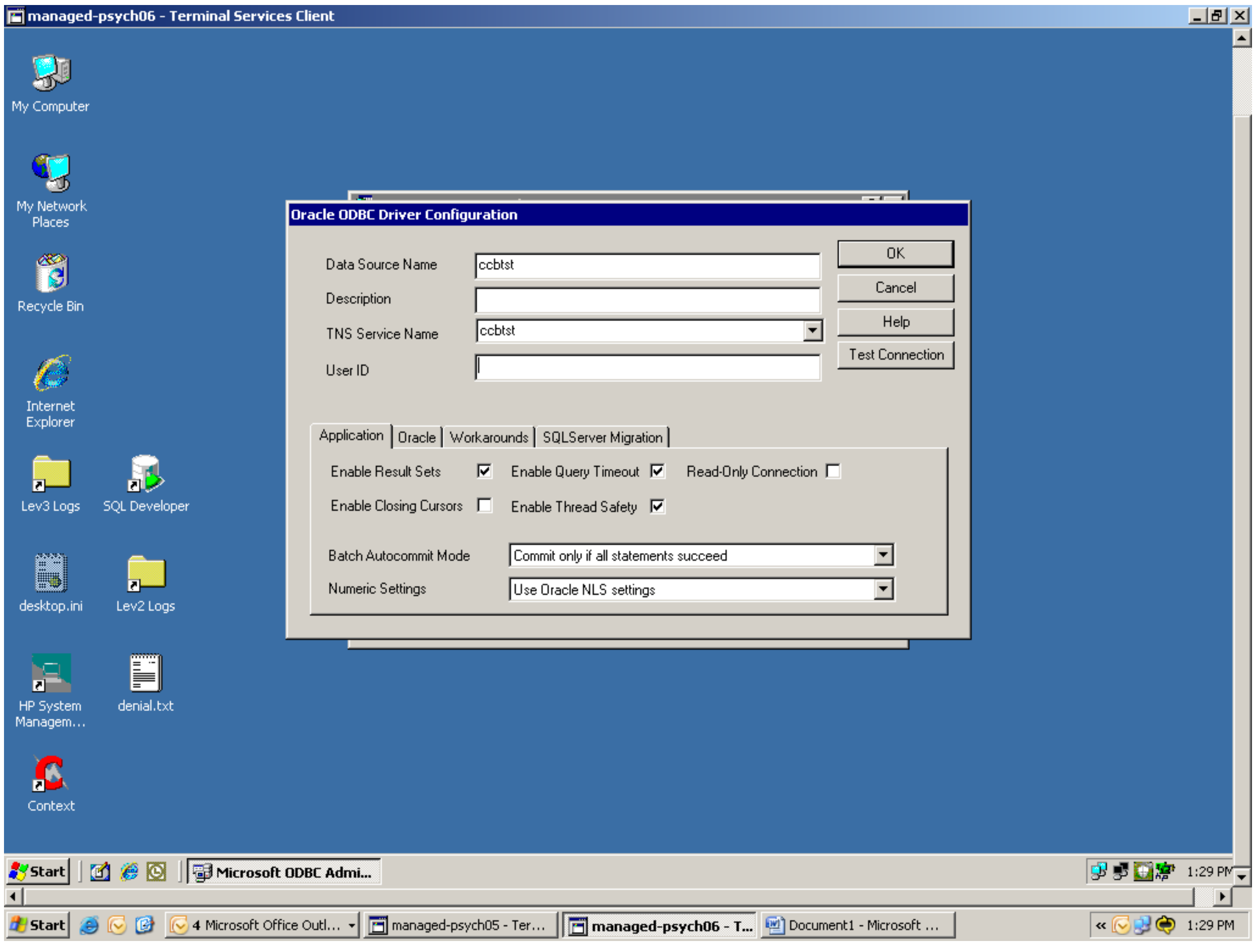
Step 2: Click on the System DSN tab and click the Add button.


managed-psych06 - Terminal Services Client
ODBC Data Source Administrator
Create New Data Source
Back
Finish
Cancel
Microsoft ODBC Admi...
1:29 PM

Select a driver for which you want to set up a data source.

Name	V
Microsoft Paradox-Treiber (*.db)	4.
Microsoft Text Driver (*.txt; *.csv)	4.
Microsoft Text-Treiber (*.txt; *.csv)	4.
Microsoft Visual FoxPro Driver	1.
Microsoft Visual FoxPro-Treiber	1.
Oracle in OraDb10g_home1	1
SAS	9.
SQL Native Client	2
SQL Server	2

Step 3: Click on the ODBC driver you want to use for your configuration, in my example Oracle.



Oracle ODBC Driver Configuration

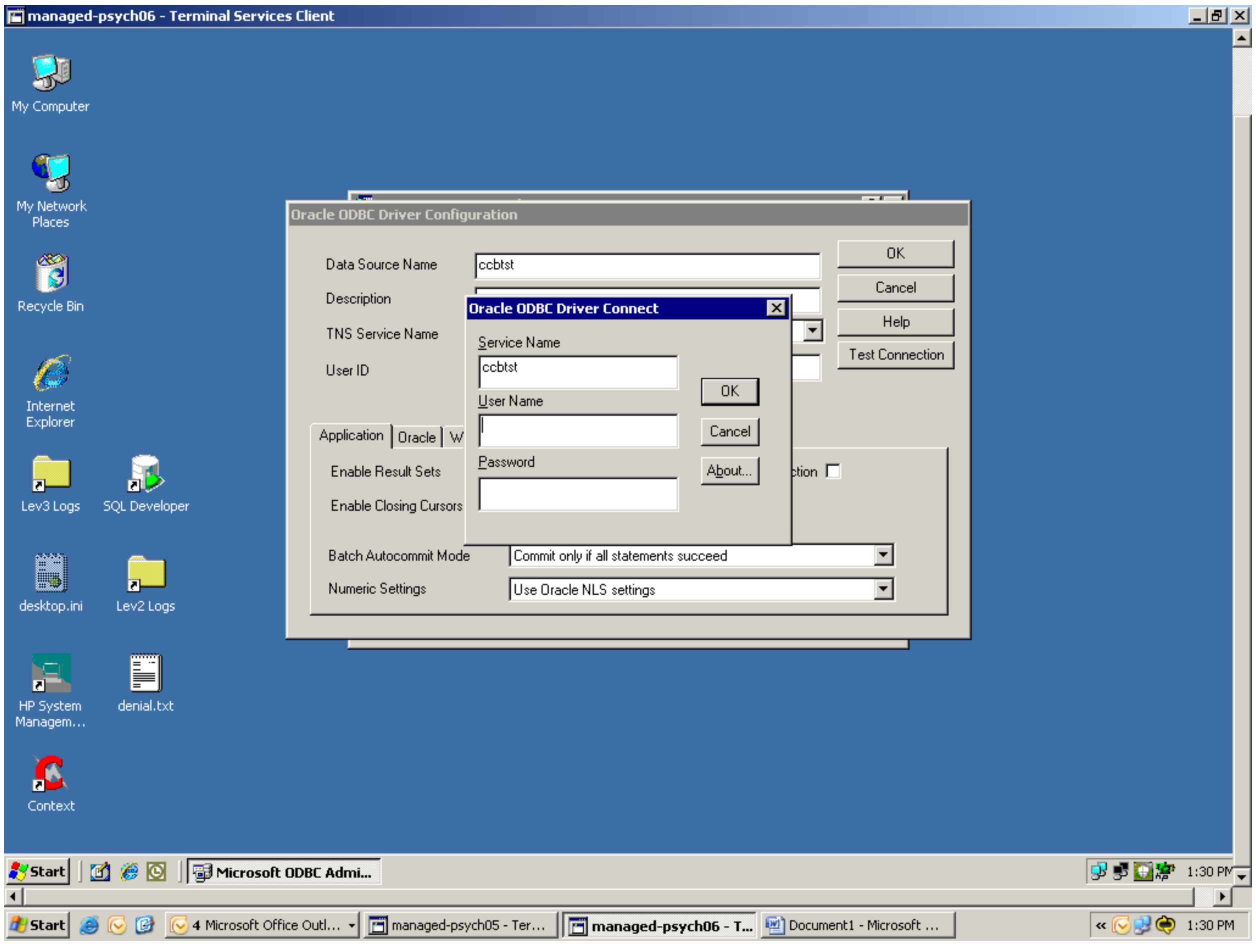
Data Source Name: ccbtst
Description:
TNS Service Name: ccbtst
User ID:

Application | Oracle | Workarounds | SQLServer Migration

Enable Result Sets ✓ Enable Query Timeout ✓ Read-Only Connection ☐
Enable Closing Cursors ☐ Enable Thread Safety ✓

Batch Autocommit Mode: Commit only if all statements succeed
Numeric Settings: Use Oracle NLS settings

Step 4: Give your data source a name. For Oracle, you will need a tnsnames.ora file and a TNS service name. If you don’t know this your DBA or system admistrator will be able to help you.


Oracle ODBC Driver Configuration

Data Source Name: ccbtst

Oracle ODBC Driver Connect

Service Name: ccbtst

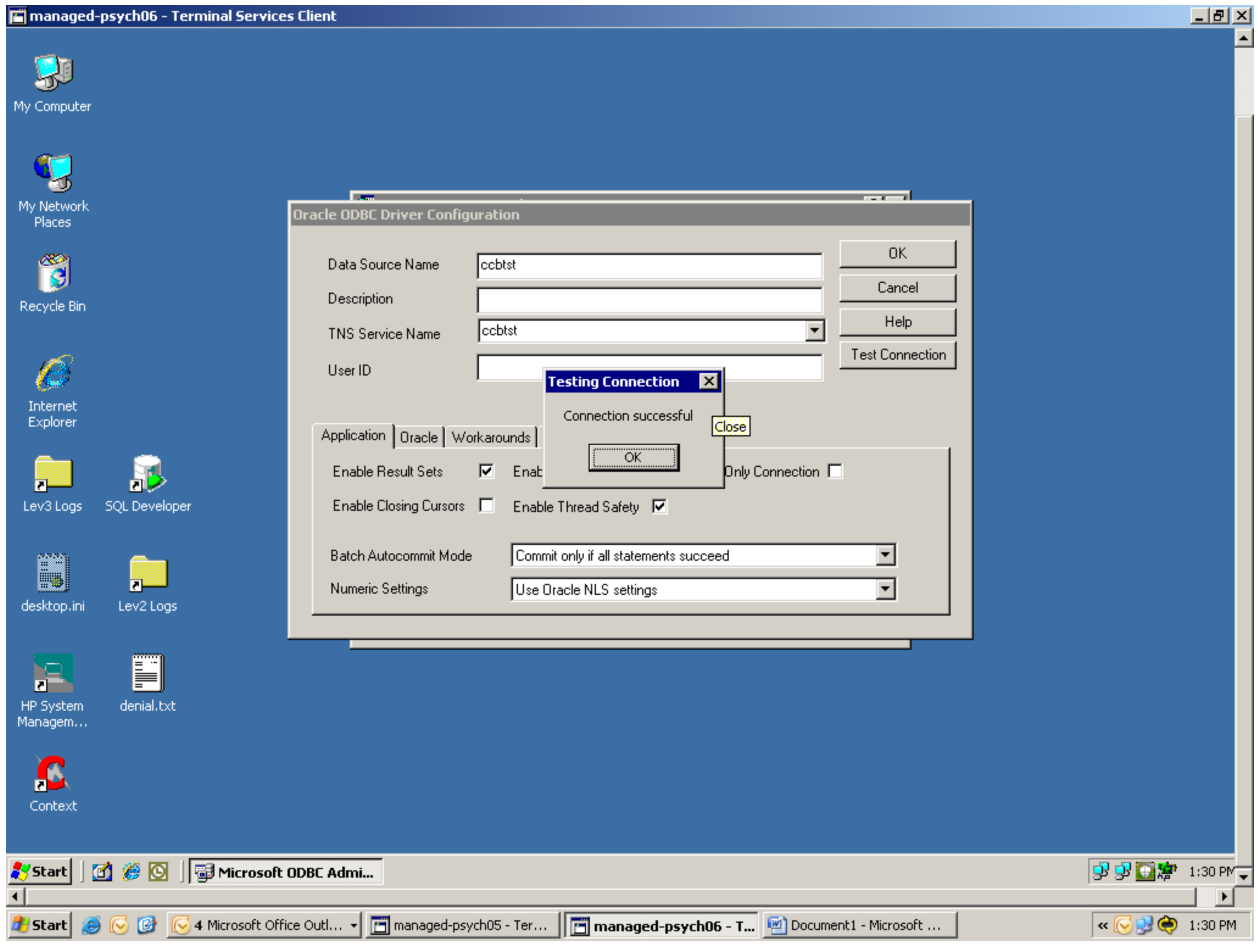
User Name:

Password:

Batch Autocommit Mode: Commit only if all statements succeed

Numeric Settings: Use Oracle NLS settings

Step 5: Click the Test Connection button and use the appropriate user name and password to test your connection to the data source.



Oracle ODBC Driver Configuration

Data Source Name: ccbtst
Description:
TNS Service Name: ccbtst
User ID:

Testing Connection
Connection successful
OK

Enable Result Sets ✓
Enable Closing Cursors ☐
Enable Thread Safety ✓
Only Connection ☐
Batch Autocommit Mode: Commit only if all statements succeed
Numeric Settings: Use Oracle NLS settings

Step 6: You should receive a Connection Successful message.

Make note of the Data Source Name you assigned, in my example, ccbtst. You will need this to define your SAS library.

To define an ODBC library in SAS to the Oracle database uses the following syntax that includes the Data Source Name you defined. The syntax is short and sweet because you created the ODBC definition above. And you are accessing the tnsnames.ora file supplied by the DBA.

```
LIBNAME DSNLIB ODBC DSN=CCBTST USER=LIOTUSLG PASSWORD=*****;
```

GET CONNECTED WITHOUT USING GUI

If you don't want to go through the ODBC configuration tool to define the System DSN as we did above, you can attach to your database using what I call a dsn-less connection. You will have to provide all of the connection information on the LIBNAME line statement. You will need the exact name of the ODBC driver to include in the LIBNAME statement. This type of statement, like the ODBC configuration, requires the tnsnames.ora file.

```
LIBNAME DSNLESS ODBC NOPROMPT="DRIVER={ORACLE IN
ORADB10G_HOME1};DBQ=CCBTST;UID=LOGIN;PWD=*****";
```

Another way to attach to an Oracle database using ODBC is without ODBC definition and without the tnsnames.ora file I mentioned above. Building on what we have been doing with the definitions and LIBNAME statement, the less predefined in ODBC or tnsnames.ora, the more needed on the LIBNAME statement. You are basically including all of the ODBC information and the tnsnames.ora information on the LIBNAME statement. This is what I call a tns-less (and dsn-less) definition.

```
LIBNAME TNSLESS ODBC NOPROMPT="DRIVER=Microsoft ODBC for Oracle;
SERVER=(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=C9DEV01.ISDIP.UPMC.EDU)(PORT =
1521))(CONNECT_DATA=(SERVICE_NAME = CCBTST.WORLD)));UID=LOGIN; PWD=*****";
```

Here is an example of a dns-less connection to a Microsoft SQL Server database. Again, you need to know the exact name of the driver. If you define the ODBC data source for the Microsoft SQL Server database, this line will be shorter.

```
LIBNAME DNSLESS ODBC NOPROMPT="DRIVER=SQL SERVER; SERVER=MP18; UID=LOGIN;
PWD=*****;DATABASE=PCP" SCHEMA=DBO;
```

The example below is another ODBC dsn-less LIBNAME but attaching to a Microsoft Access database.

```
LIBNAME DSNLESS ODBC NOPROMPT="DRIVER={MICROSOFT ACCESS DRIVER (*.MDB)};DBQ=\\MANAGED-
PSYCH06\C$\CODE-LIBRARY\LIOTUSLG\PERSONAL\DATABASE1.MDB";
```

The final LIBNAME syntax examples I will show pertains to another connection engine, SAS/ACCESS for OLE DB. An advantage to OLE DB is that the password can be encrypted.

This example connects to a Microsoft SQL Server database.

```
LIBNAME PREC04 OLEDB PROVIDER=SQLOLEDB PROPERTIES='INITIAL CATALOG'=THEPRECIOUS
DATASOURCE='MANAGED-PSYCH18' SCHEMA=DBO  USERID=LIOTUSLG
PASSWORD="{SAS001}ZHVUZWRPBG==";
```

This example is for a Microsoft Excel spreadsheet. This connection string will create a separate table for each worksheet or tab contained in the spreadsheet.

```
LIBNAME MISCSRC1 OLEDB  PROVIDER_STRING='EXCEL 8.0;IMEX=1'  DATASOURCE='C:\CODE-
LIBRARY\SAS_ETL\DATA\DW_PROTOTYPE\DWPT1_MASTER_MISC_SRC.XLS'  PROVIDER=JET  PROMPT=NO
;
```

One final connectivity example I would like to provide is SQL Pass-through using SAS/ACCESS Interface to ORACLE. Sometimes it is necessary to let the database handle the sorting, joins etc. When you have large datasets that reside in the database, it can be more advantageous to run these queries in the database thus taking advantages of any indexes that might be available on the table to make the query run faster. SQL Pass-through is another connectivity that allows for the encrypted password, a definite advantage when coding and storing scripts.

```
PROC SQL;

CONNECT TO ORACLE AS ORACLAIMS(PATH=CCBTST  USER=LIOTUSLG  PASSWORD="{SAS001}CM9JA3K="
CONNECTION=GLOBAL);

CREATE VIEW WORK.MAPPED AS  SELECT * FROM CONNECTION TO ORACLAIMS((SELECT
EOP_CODE_ID_CC,EOP_CODE FROM EOP_REF_CODE);

QUIT;
```

CONCLUSION

The examples above are just a smattering of what can be done with the various SAS/ACCESS libraries. They just happen to be what I have needed for our various applications and users. If you have a Citrix environment and want to use the DSN ODBC connection, the Citrix administrator will have to define the ODBC data source for your use. When using a dns-less connection, regardless if using SAS via a Citrix environment, workstation or server, you will need to know the exact name of the driver being used to connect to the application. If you use Oracle, you will need to work with the DBA or system administrator to obtain the tnsnames.ora file or the information contained in this file. As far as using the DSN or dsn-less connection or connection with the tnsnames information embedded, it all depends on how much set-up you want to perform. Oftentimes, the technical level of the user is the driving force.

I hope this information is useful. I found this information in a multitude of places so hoped that by consolidating the information in one place, SAS programmers can build on this information based on their application needs and associated SAS/ACCESS engines.

A follow-up paper with this same type of connectivity information using DI Studio is forthcoming.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Laura Liotus
Community Care Behavioral Health
One Chatham Center
Suite 700
Pittsburgh, PA 15219
412-402-8713
liotuslg@upmc.edu
http://www.ccbh.com/