Accessing a Microsoft SQL Server Database from SAS on Microsoft Windows

On Microsoft Windows, you have two options to access a Microsoft SQL Server database from SAS. You can use either SAS/Access Interface to ODBC, or SAS/Access to OLEDB.

Submitting the following code from within SAS displays all the licensed products for your site in the SAS log window:

Proc setinit noalias;
Run;

If you have one or both of the Access products licensed for your site, the next step is to determine if the products have been installed on your machine.

From Windows Explorer, you can browse to !SASROOT\Access\Sasexe and look for the following files:

1) **sasioodb.dll** – the presence of this file means that SAS\Access Interface to ODBC is installed on your machine.
2) **sasioole.dll** - the presence of this file means that SAS\Access Interface to OLEDB is installed on your machine.

Depending on how SQL Server is set up, you can connect using either SQL Server Authentication or NT Authentication. Using SAS/Access to ODBC or SAS/Access to OLEDB with each authentication method will be discussed below.
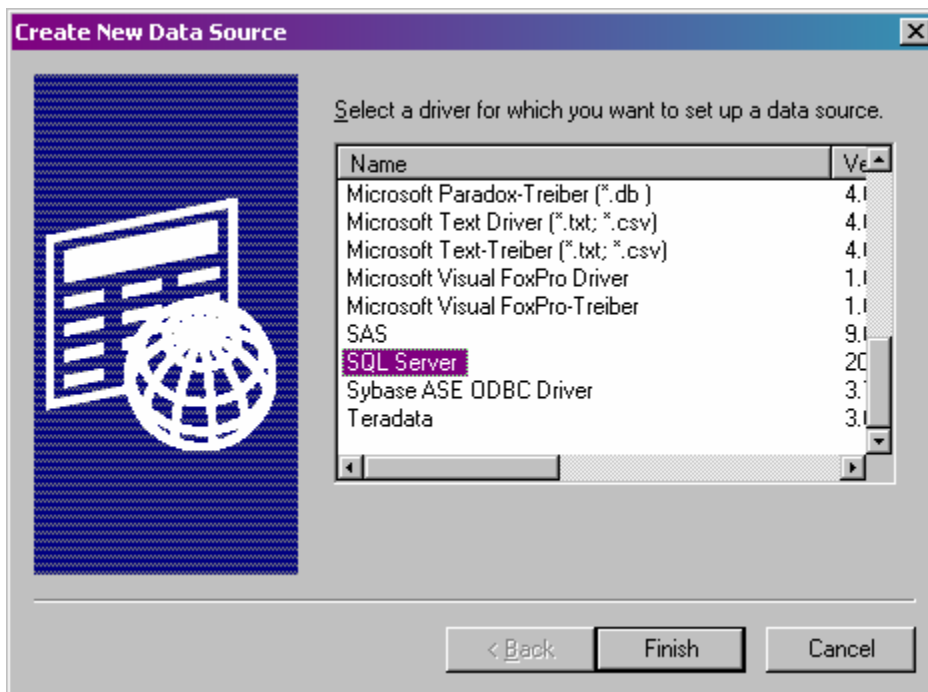
## SAS/Access to ODBC:

SQL Server Authentication:

To set up an ODBC Data Source, select the Start Menu, click on Settings → Control Panel and choose Administrative Tools. From there, choose Data Sources (ODBC). This will open the Data Source Administrator.
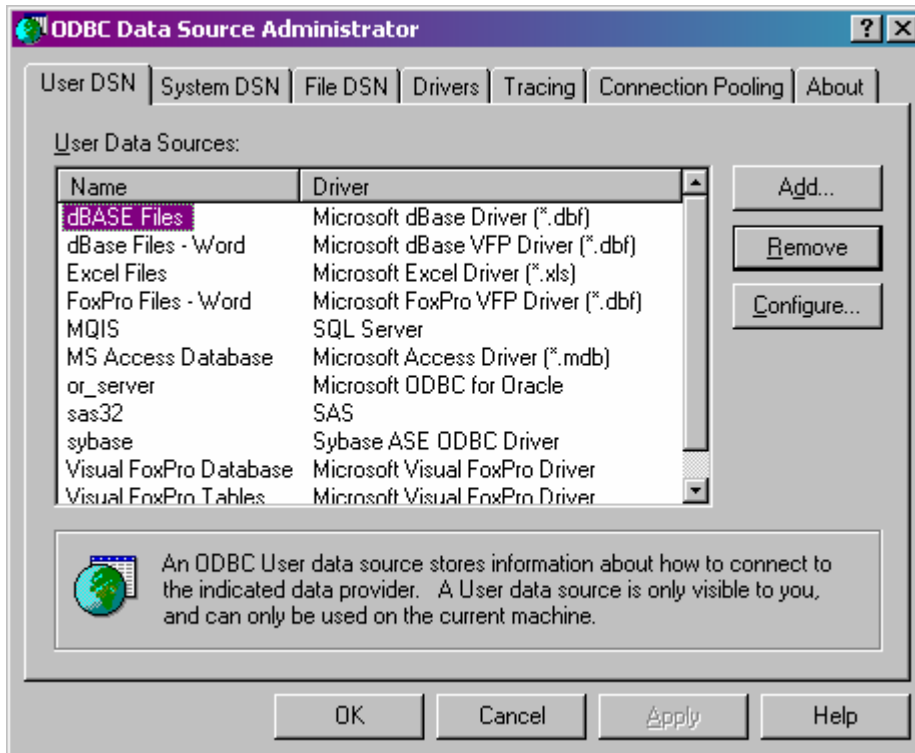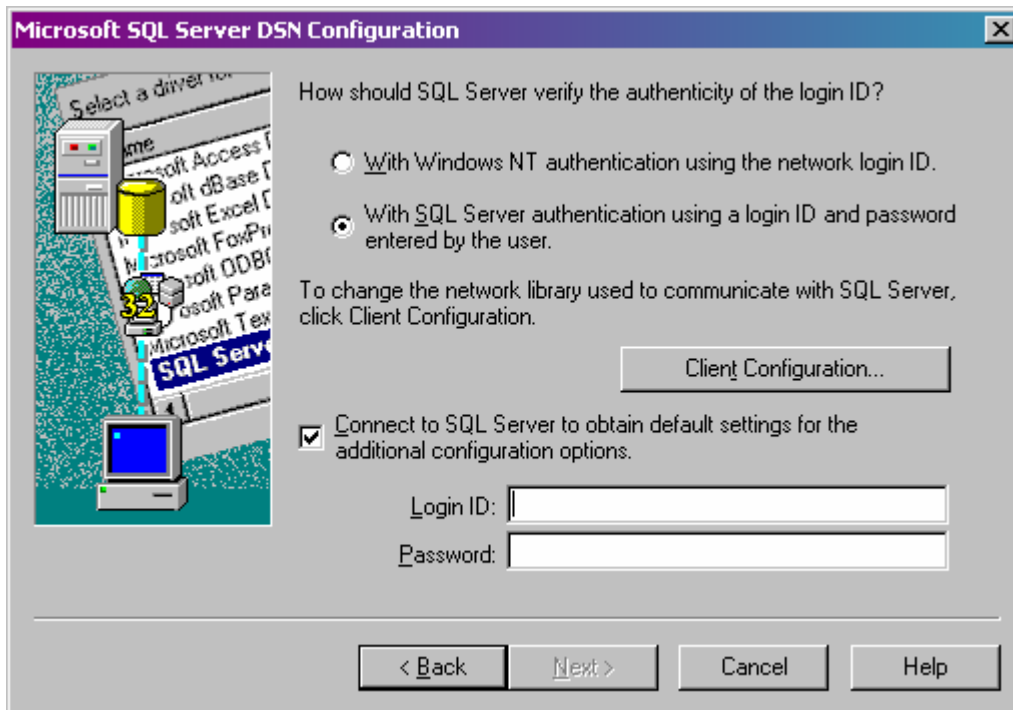
Choose the User or System DSN tab and select 'Add' to add a new data source. Select the SQL Server driver and click 'Finish'.
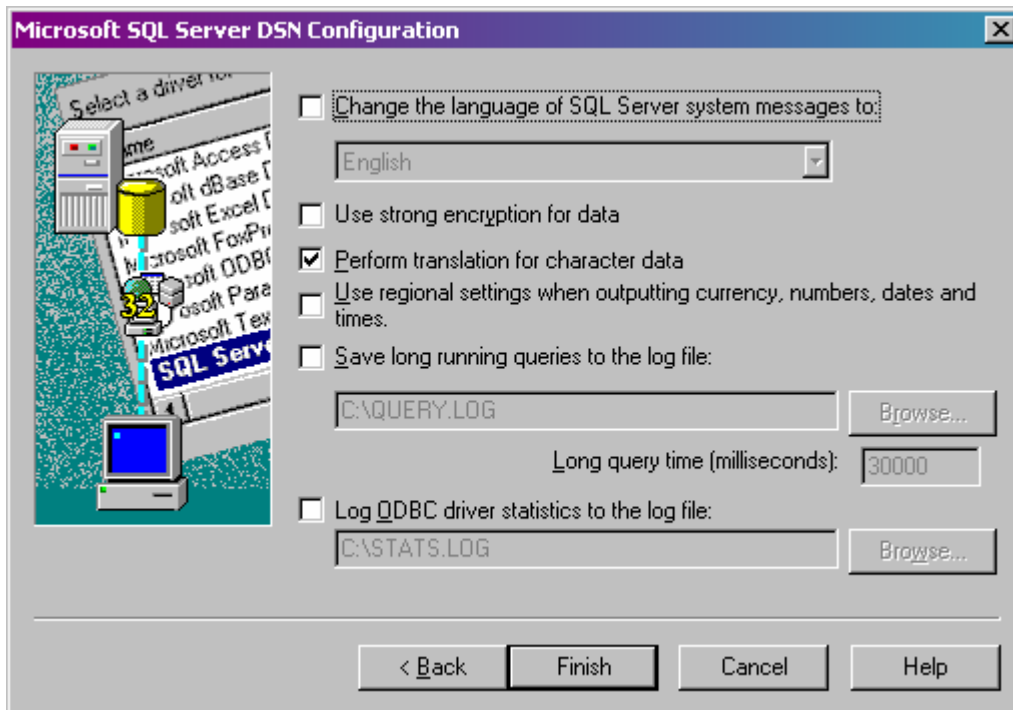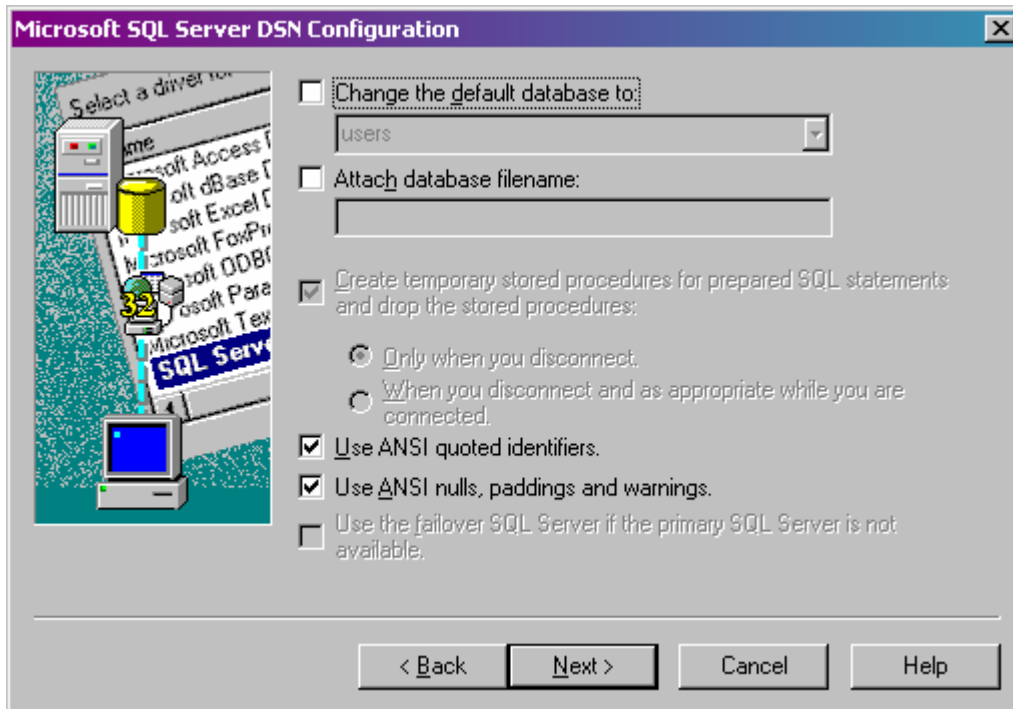

The next window allows you to enter a name for the data source, an optional description, and the server you want to connect to.
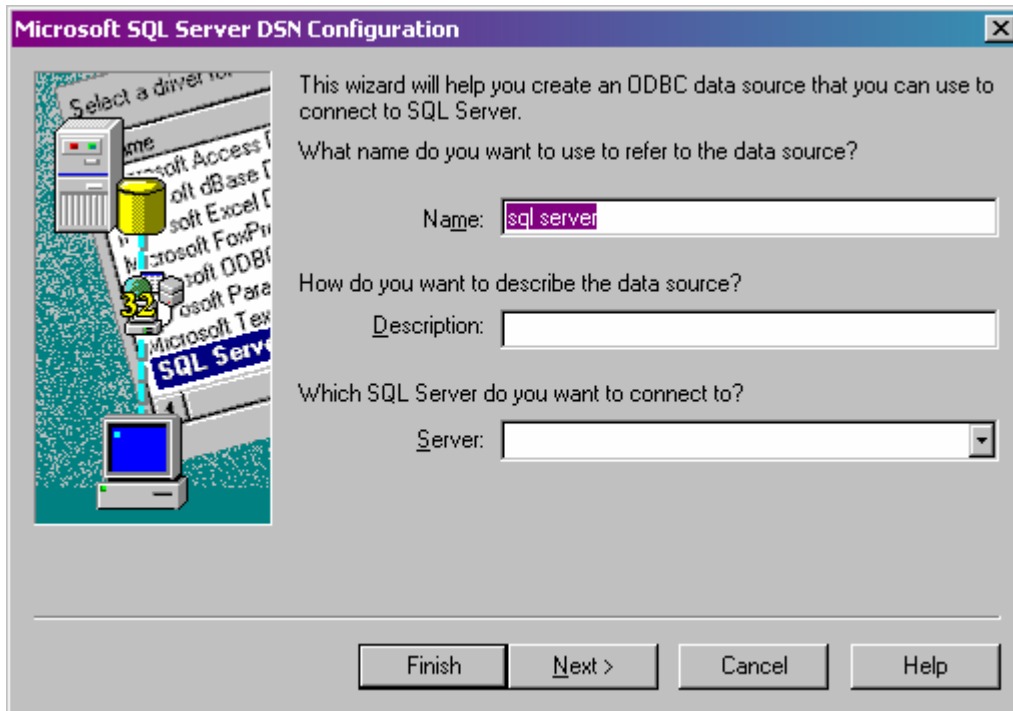
Choose SQL Server authentication as shown below and enter the SQL server login ID and password. Click on 'Next'.

The next 2 screens will allow for further server configurations, such as changing the default database, creating temporary stored procedures, changing the language of SQL server system messages, etc.

Once you click the 'Finish' button, you will see a summary window of the configurations you chose, and you can try a test connection to see if the configurations are valid.

ODBC Microsoft SQL Server Setup

A new ODBC data source will be created with the following configuration:

```
Microsoft SQL Server ODBC Driver Version 03.81.9030

Data Source Name: sql server
Data Source Description:
Server: lupin\sql2000
Database: (Default)
Language: (Default)
Translate Character Data: Yes
Log Long Running Queries: No
Log Driver Statistics: No
Use Integrated Security: No
Use Regional Settings: No
Prepared Statements Option: Drop temporary procedures on
disconnect
Use Failover Server: No
Use ANSI Quoted Identifiers: Yes
Use ANSI Null, Paddings and Warnings: Yes
Data Encryption: No
```
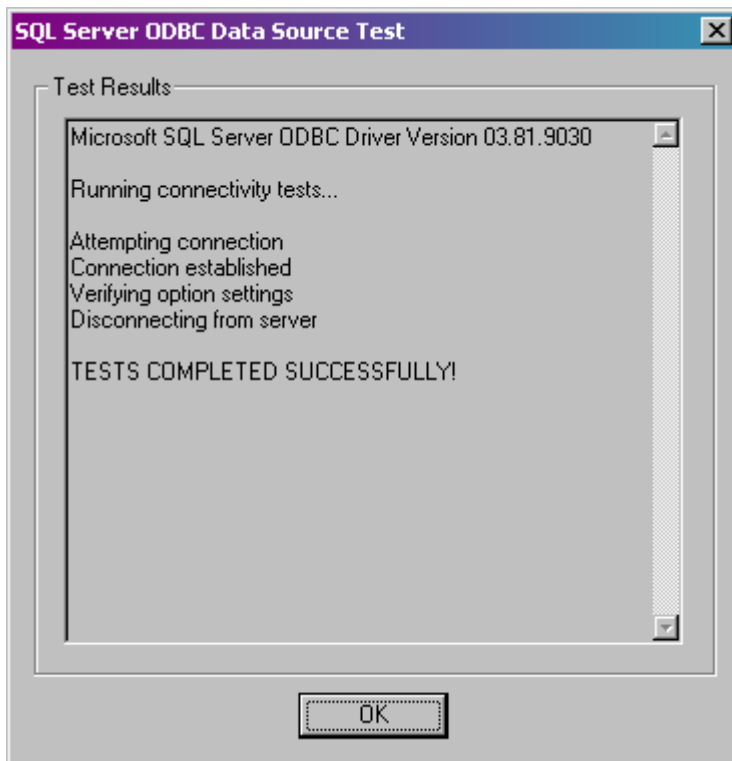
Test Data Source...   OK   Cancel

If everything is set up properly, the test connection will be successful. Click 'OK' to exit out of the SQL Server setup and Administrator.

SQL Server ODBC Data Source Test

Test Results

```
Microsoft SQL Server ODBC Driver Version 03.81.9030

Running connectivity tests...

Attempting connection
Connection established
Verifying option settings
Disconnecting from server

TESTS COMPLETED SUCCESSFULLY!
```

OK

Once the driver has been configured and the test connection is successful, then you can use a LIBNAME statement to create a library within SAS:

LIBNAME SQL ODBC DSN='sql server' user=sasjlb pw=pwd;

Where 'sql server' is the name of the Data Source configured in the ODBC Administrator.
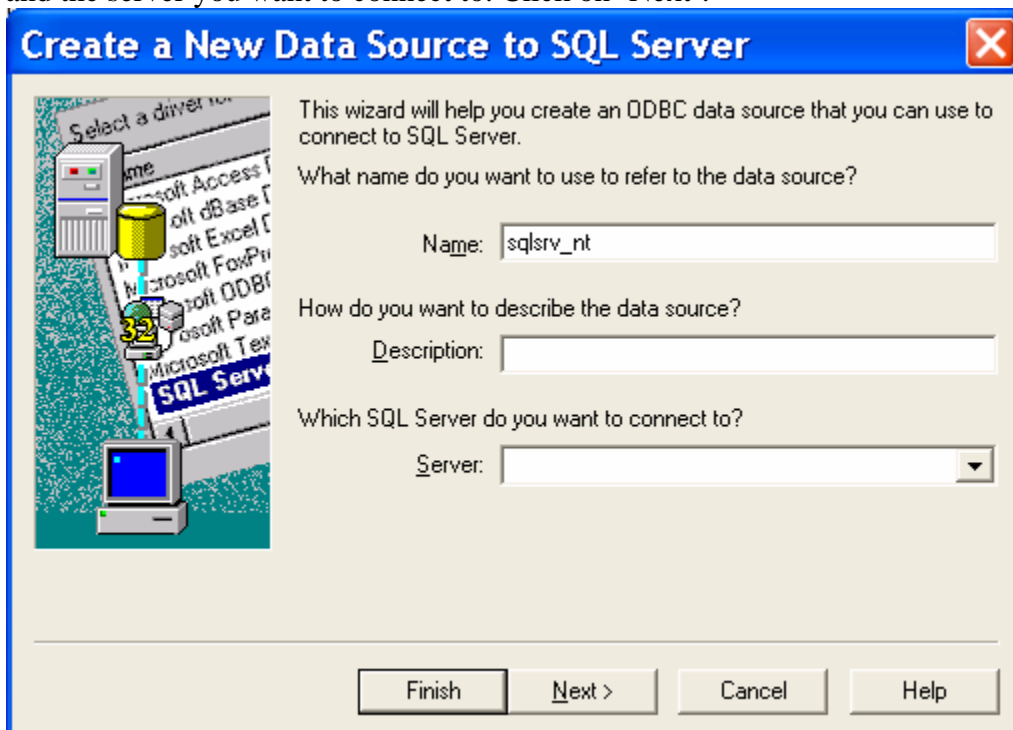
Using NT Authentication:

To setup an ODBC data source, select the Start Menu, click on Settings → Control Panel and choose Administrative Tools. From there, choose Data Sources (ODBC). This will open the Data Source Administrator.
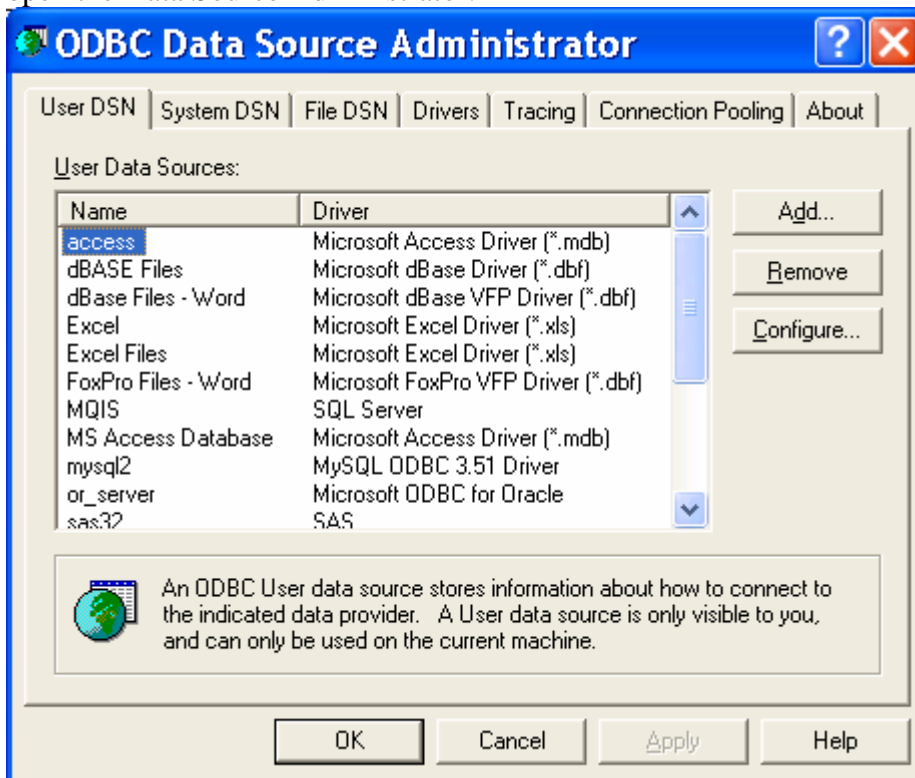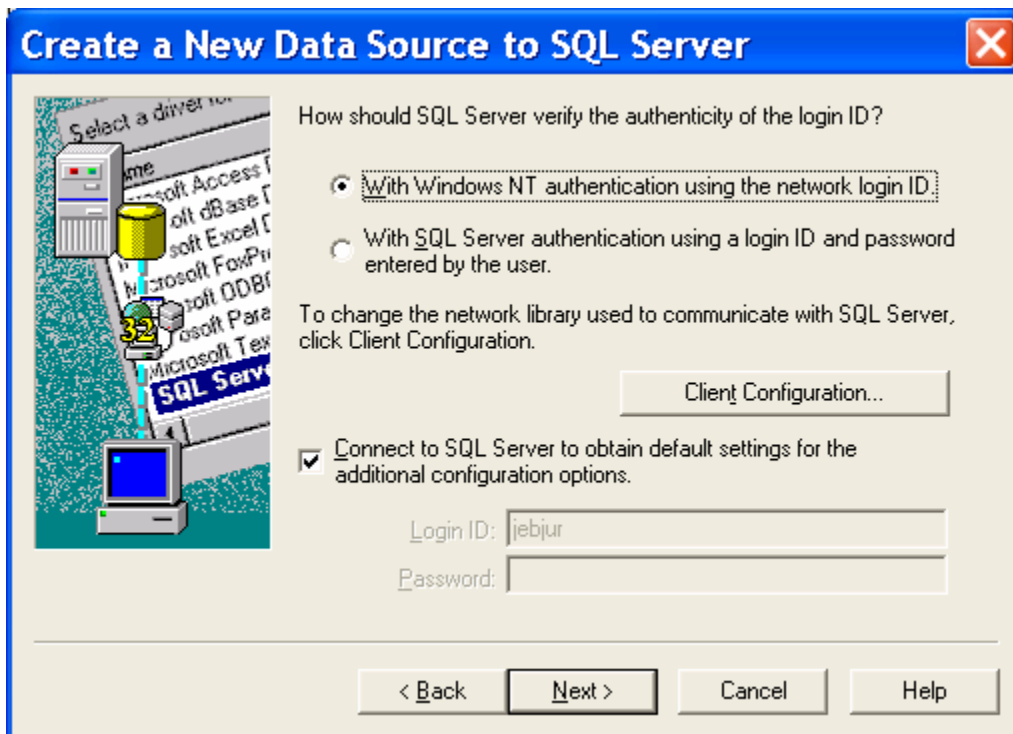

Choose the User or System DSN tab and select 'Add' to add a new data source. Select the SQL Server driver and click 'Finish'.
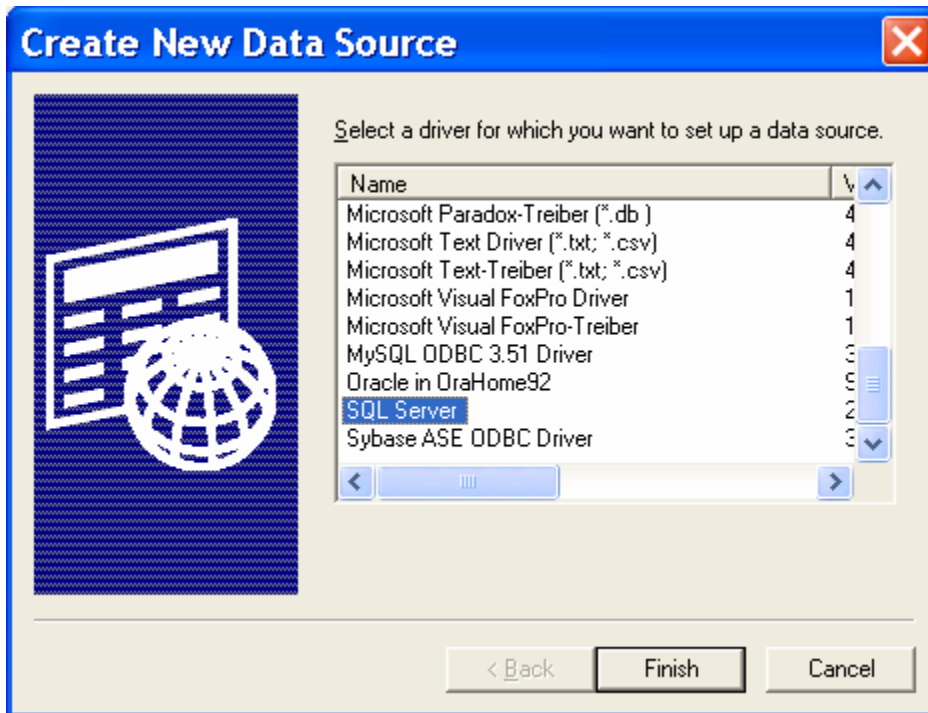
The next window allows you to enter a name for the data source, an optional description, and the server you want to connect to. Click on 'Next'.

Choose NT authentication as shown below and click on 'Next'.
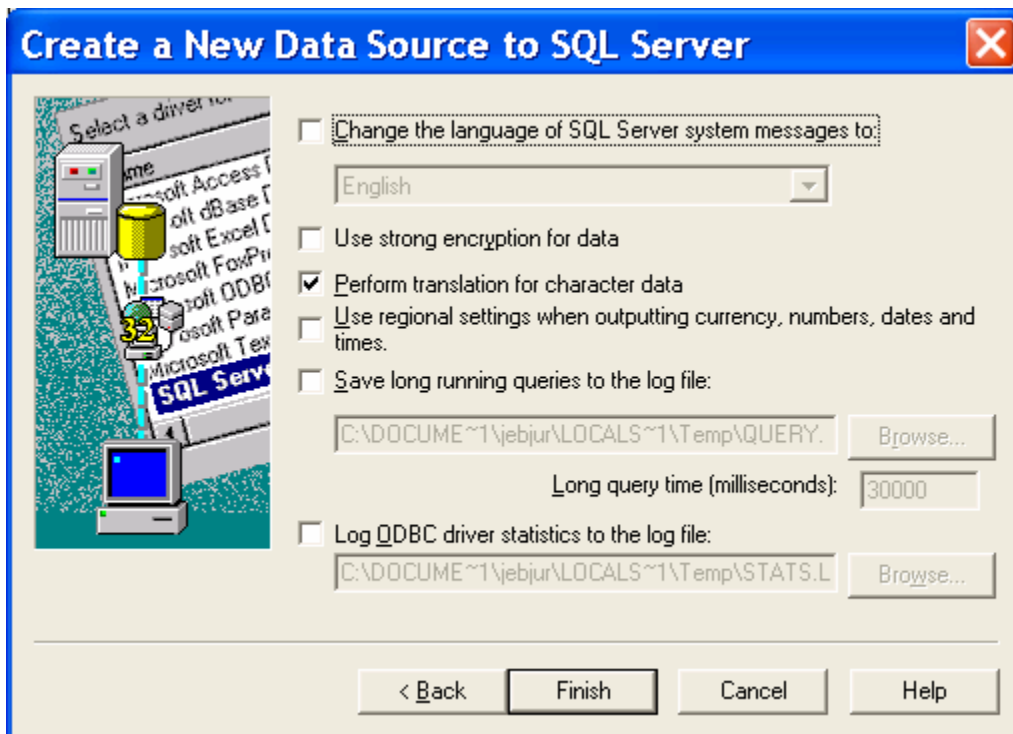
The next 2 screens will allow for further server configurations, such as changing the default database, creating temporary stored procedures, changing the language of SQL server system messages, etc.
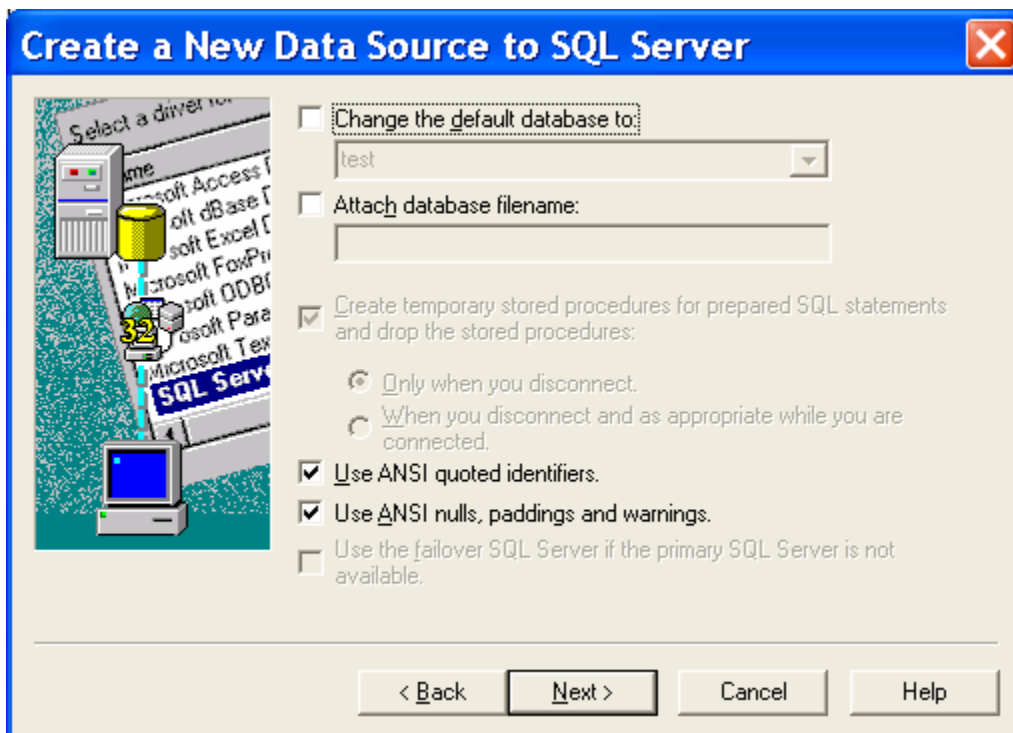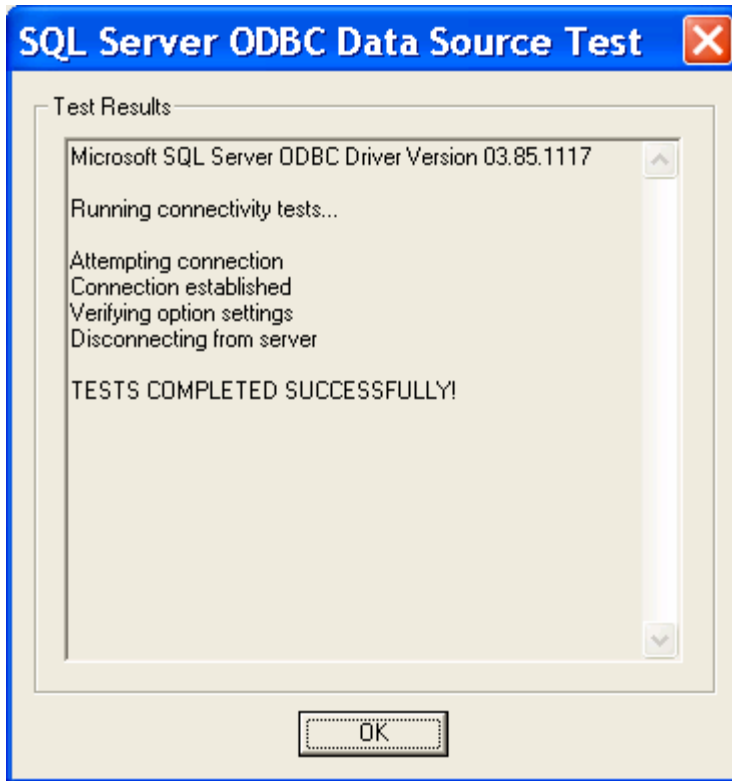
Once you click the 'Finish' button, you will see a summary window of the configurations you chose, and you can try a test connection to see if the configurations are valid.

If everything is set up properly, the test connection will be successful. Click 'OK' to exit out of the SQL Server setup and Administrator.

Once the driver has been configured and the test connection is successful, then you can use a LIBNAME statement to create a library within SAS:

LIBNAME SQL ODBC DSN='sqlsrv_nt';

Where 'sqlsrv_nt' is the name of the Data Source configured in the ODBC Administrator.

Prompted connection:

If you aren't sure what values to add for the userid, password and data source, you can try connecting with a prompted connection. A prompted connection just means you are prompted to enter the above information instead of supplying it on the LIBNAME statement. Submit the following 2 lines of code:

```
libname sql odbc prompt;
%put %superq(sysdbmsg); /* V9 syntax */
%put &sysdbmsg;          /* V8 syntax */
```

The 'Select Datasource' window will open. If you created a user or system DSN, you will need to select the Machine Data Source tab. Choose the appropriate DSN, and click on

'Okay'. Enter your SQL Server login id and password. Once you have connected, several parameters will be written to the log window. An example is below:

```
LIBNAME SQL ODBC prompt;
NOTE: Libref SQL was successfully assigned as follows:
      Engine:         ODBC
      Physical Name: sqlsrv
7     %put %superq(sysdbmsg);
ODBC: DSN=sqlsrv;UID=jebjur;PWD=jebjur1;WSID=d17117
```

You can cut and paste everything after 'ODBC:' and place it on the LIBNAME statement with a NOPROMPT= option as such:

/* SQL Server Authentication */

LIBNAME SQL ODBC noprompt= "dsn=sqlsrv; uid=sasjlb; pwd=pwd; wsid=d17117";

/* NT Authentication */

LIBNAME SQL ODBC noprompt="dsn=sqlsrv_nt;wsid=d17117;
Trusted_Connection=Yes ";

Schemas:

SQL Server database tables are organized in schemas, which are equivalent to database users or owners. In order to see particular tables in a defined library, you may need to add the SCHEMA= option to the LIBNAME statement. If no schema is specified, SAS will look in the current userid's schema by default.

For example:

LIBNAME SQL ODBC DSN=sqlsrv user=sasjlb pw=pwd schema=dbo;

If you aren't sure what schema your tables are contained in, you can use one of the following methods to find it:

1) Use SAS Query Window: from the Tools -> Query menu.

when the Query Window has loaded, go to Tools – Switch Access Mode - ODBC. Then select your datasource and respond to any prompts that pop up. When you are connected, you will see a list of available tables from your odbc datasource. The tables are two-level names such as dbo.table1. The first level (dbo) is the schema.

2) Use PROC SQL passthrough method:

This method will create a temporary data set with the list of available tables in the database. The TABLE_SCHEM variable will contain the schema.

```
/* SQL Server Authentication */

proc sql;
connect to odbc (dsn=sqlsrv  user=user pwd=xxxxx);
create table test as select * from connection to odbc(ODBC::SQLTables);
quit ;

/* NT Authentication */

proc sql;
connect to odbc (dsn='sqlsrv_nt');
create table test as select * from connection to odbc(ODBC::SQLTables);
quit ;
```

Importing Data:

Once the LIBNAME statement has been successfully assigned, you can use either DATA step or PROC SQL logic to import the data in a SQL Server table, just as you would with a permanent SAS data set.

For ex:

```
LIBNAME SQL ODBC DSN='sql server' user=sasjlb pw=pwd;

DATA NEW;
SET SQL.TABLE1;
RUN;

PROC SQL;
CREATE TABLE NEW AS SELECT * FROM SQL.TABLE1;
QUIT;
```

You can also use PROC SQL passthrough with SAS/Access to ODBC. The query would look similar to:

```
/* SQL Server Authentication */

proc sql;
connect to odbc (dsn=sqlsrv  user=user pwd=xxxxx);
create table test as select * from connection to odbc(select * from table2);
quit ;

/* NT Authentication */
```

```
proc sql;
connect to odbc(dsn='sqlsrv_nt');
create table new as select * from connection to odbc(select * from table2);
quit;
```

SAS/Access to OLEDB:

SQL Server Authentication:

With SAS/Access to OLEDB, you do not have to configure the data provider. The
LIBNAME statement would look similar to:

LIBNAME sqlsrv oledb init_string="Provider=SQLOLEDB.1;Password=pwd;
Persist Security Info=True;User ID=user;Data Source=sqlserv";

Using NT Authentication:

With NT authentication, The LIBNAME statement would look similar to:

LIBNAME sqlsrv oledb init_string="Provider=SQLOLEDB.1;
Integrated Security=SSPI;Persist Security Info=True;
Initial Catalog=northwind;Data Source=steak";

Prompted Connection:

If you aren't sure what values to add for the userid, password and data source
(=server name), then you can try connecting with a prompted connection. A
prompted connection just means you are prompted to enter the above information
instead of supplying it on the LIBNAME statement. Submit the following 2 lines
of code:

```
libname sqlsrv oledb;
%put %superq(sysdbmsg); /* V9 syntax */
%put &sysdbmsg;          /* V8 syntax */
```

* in the pop up window, select "Microsoft OLE DB Provider for SQL
Server,"
* select "Next:
* enter the Data Source name (server)
* select the "Use a specific user name and password" radio button, and
  enter the appropriate username and password.
* Enter the name of a database on the server you wish to connect to (optional).
* select "Test Connection" and made sure it established a connection
* select "OK" to exit the pop up. If a connection was established, you

should see a note in the SAS log that says the LIBNAME statement was successfully assigned.

If you then want to do an unprompted connection, then do the following:

* The connection parameters were then written to the log due to the following line:
%put %superq(SYSDBMSG);

* which wrote the following to the Log:

OLEDB: Provider=SQLOLEDB.1;Password=pwd;Persist Security Info=True; User ID=user;Data Source=sqlserv

In order to create a LIBNAME statement such as the one above, you can cut and paste the connection parameters that were written to the log (everything after the OLEDB: string) and add them to the LIBNAME statement with an INIT_STRING= option. The final LIBNAME statement looks like:

/* SQL Server Authentication */

LIBNAME sqlsrv oledb init_string="Provider=SQLOLEDB.1;Password=pwd; Persist Security Info=True;User ID=user;Data Source=sqlserv";

/* NT Authentication */

LIBNAME sqlsrv oledb init_string="Provider=SQLOLEDB.1;Integrated Security=SSPI; Persist Security Info=True;Initial Catalog=northwind;Data Source=steak";

If the connection is successful, you can go to the SAS Explorer window, click on the library and see the tables on the server.

Schemas:

If the LIBNAME statement connected successfully but there are no tables in the library, a schema may be needed on the LIBNAME statement as well. If you need to find a schema for a table with SAS/Access to OLEDB, you can use the PROC SQL passthrough code below. This method will create a temporary data set with the list of available tables in the database. The TABLE_SCHEMA variable will contain the schema.

proc sql;
connect to oledb;
create table tabs as select * from connection to oledb(OLEDB::Tables);
quit;

Once you find the appropriate schema value, you would add it to the the LIBNAME statement with the SCHEMA= option. The LIBNAME statement would look similar to the following:

/* SQL Server Authentication */

LIBNAME sqlsrv oledb init_string="Provider=SQLOLEDB.1;Password=pwd;
Persist Security Info=True;User ID=user;Data Source=sqlserv" schema=dbo;

/* NT Authentication */

LIBNAME sqlsrv oledb init_string="Provider=SQLOLEDB.1;Integrated Security=SSPI;
Persist Security Info=True;Initial Catalog=northwind;Data Source=steak" schema=dbo;

Importing Data:

 Once the LIBNAME statement has been successfully assigned, you can use either DATA step or PROC SQL logic to import the data in a SQL Server table, just as you would with a permanent SAS data set.

For ex:

LIBNAME sqlsrv oledb init_string="Provider=SQLOLEDB.1;Password=pwd;
Persist Security Info=True;User ID=user;Data Source=sqlserv"

DATA NEW;
SET SQLSRV.TABLE1;
RUN;

PROC SQL;
CREATE TABLE NEW AS SELECT * FROM SQLSRV.TABLE1;
QUIT;

You can also use PROC SQL passthrough with SAS/Access to OLEDB. The query would look similar to:

/* SQL Server Authentication */

proc sql;
connect to oledb (init_string=" Provider=SQLOLEDB.1;Password=pwd;
Persist Security Info=True;User ID=user;Data Source=sqlserv" schema=dbo);
select * from connection to oledb (select * from class);
quit;

/* NT Authentication */

```
proc sql;
connect to oledb (init_string="Provider=SQLOLEDB.1;Integrated Security=SSPI;
Persist Security Info=True;Initial Catalog=northwind;Data Source=steak" schema=dbo);
select * from connection to oledb (select * from class);
quit;
```

Resources:

SAS/Access 9.1 Interface to Relational Databases: Reference can be found at the
following URL:

http://support.sas.com/onlinedoc/913/docMainpage.jsp