

# Accessing a Microsoft SQL Server Database from SAS® under Microsoft Windows

**Release Information**

Content Version: 1.1 November 2017

(This paper replaces TS-765 released in 2006.)

**Trademarks and Patents**

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

# Contents

<b>Introduction.....</b>	<b>2</b>
<b>Deciding Which SAS/ACCESS Product to Use .....</b>	<b>2</b>
<b>SAS/ACCESS to ODBC .....</b>	<b>2</b>
SQL Server Authentication .....	2
Using NT Authentication .....	7
Prompted Connection .....	11
Schemas.....	12
Importing Data .....	12
<b>SAS/ACCESS to OLE DB .....</b>	<b>13</b>
SQL Server Authentication .....	13
Using NT Authentication .....	13
Prompted Connection .....	13
Schemas.....	14
Importing Data .....	14
<b>Resources .....</b>	<b>15</b>

## Introduction

With regards to working with relational databases in SAS®, a nice feature about SQL Server is the relatively small amount of configuration it requires to start working with the data. There is no database client software to install, and the drivers are usually installed for you (on a Windows machine). Once you configure the drivers to work with your database, you are ready to start working. This paper will describe how to configure both the SQL Server OLE DB and ODBC drivers as well as the different ways to connect to your database.

## Deciding Which SAS/ACCESS Product to Use

With 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 OLE DB.

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 SAS/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:

- **sasioodb.dll**: The presence of this file means that SAS\ACCESS Interface to ODBC is installed on your machine.
- **sasioole.dll**: The presence of this file means that SAS\ACCESS Interface to OLE DB is installed on your machine.

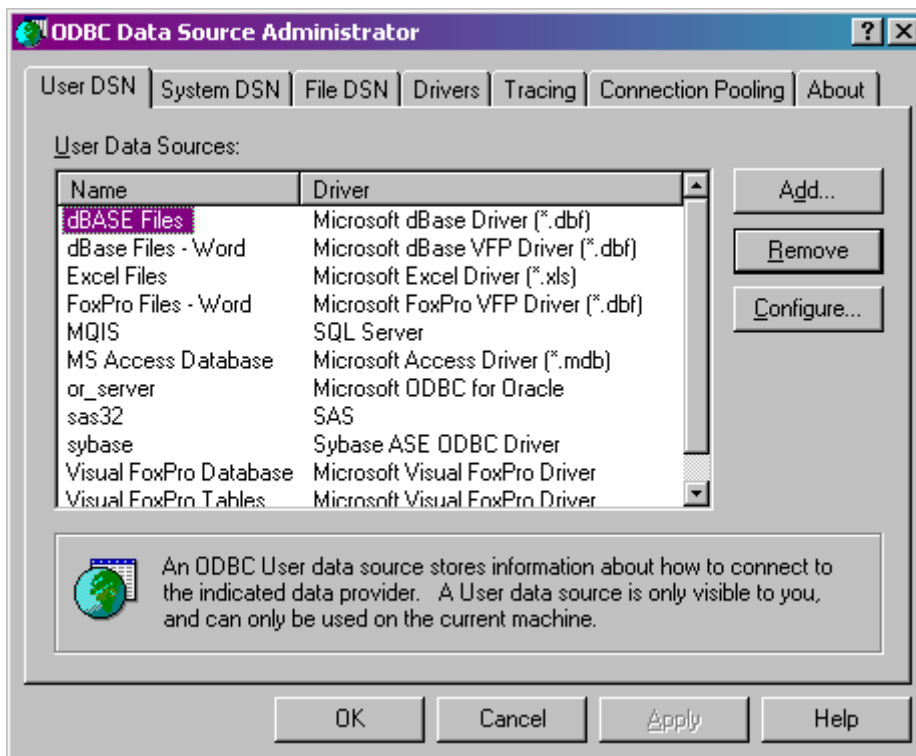
Depending on how SQL Server is set up, you can connect using either SQL Server Authentication or NT Authentication. Using SAS/ACCESS to ODBC or SAS/ACCESS to OLE DB with each authentication method is discussed below.

## SAS/ACCESS to ODBC

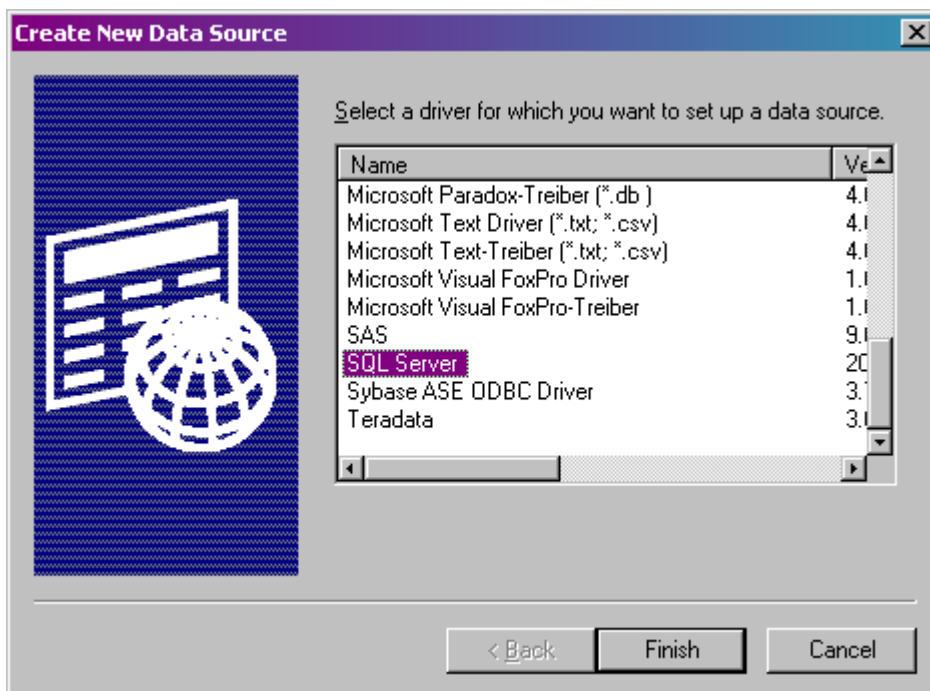
With SAS/Access to ODBC, you will configure the SQL Server ODBC driver in the Data Source Administrator to work with your SQL Server database server and tables.

## SQL Server Authentication

To set up an ODBC Data Source, select the **Start Menu**, click **Settings ► Control Panel**, and choose **Administrative Tools**. From there, choose **Data Sources (ODBC)**. This opens the ODBC Data Source Administrator dialog box.



Click the **User DSN** or the **System DSN** tab and click **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.

**Microsoft SQL Server DSN Configuration**

This wizard will help you create an ODBC data source that you can use to connect to SQL Server.

What name do you want to use to refer to the data source?

Name:

How do you want to describe the data source?

Description:

Which SQL Server do you want to connect to?

Server:

Choose SQL Server authentication as shown below and enter the SQL server login ID and password. Click **Next**.

**Microsoft SQL Server DSN Configuration**

How should SQL Server verify the authenticity of the login ID?

☐ With Windows NT authentication using the network login ID.

☒ With SQL Server authentication using a login ID and password entered by the user.

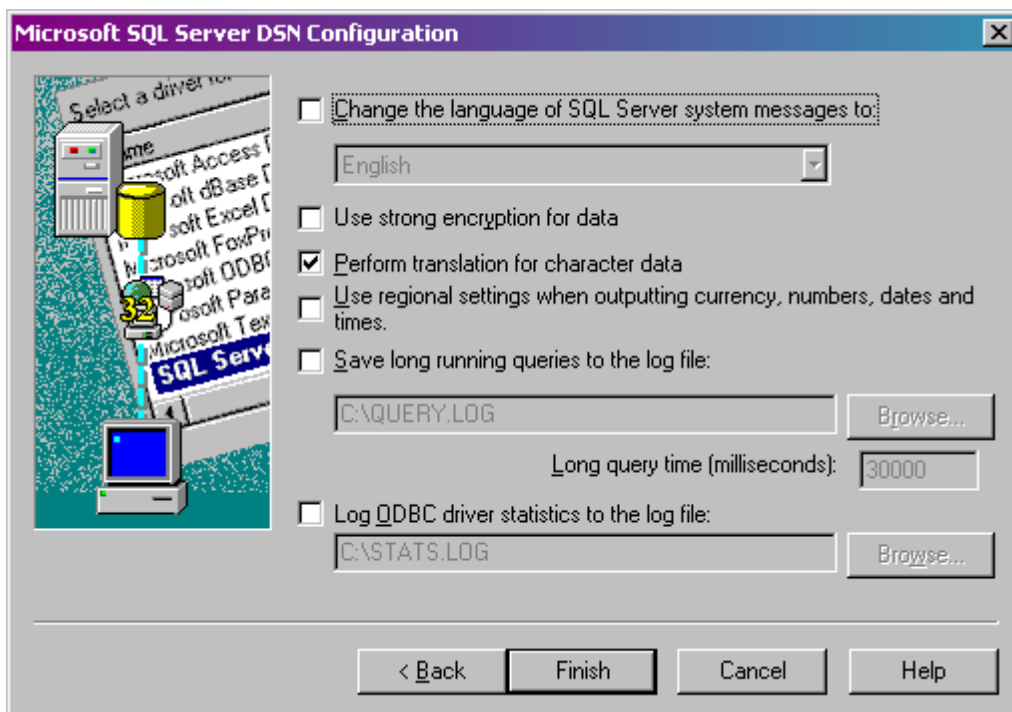
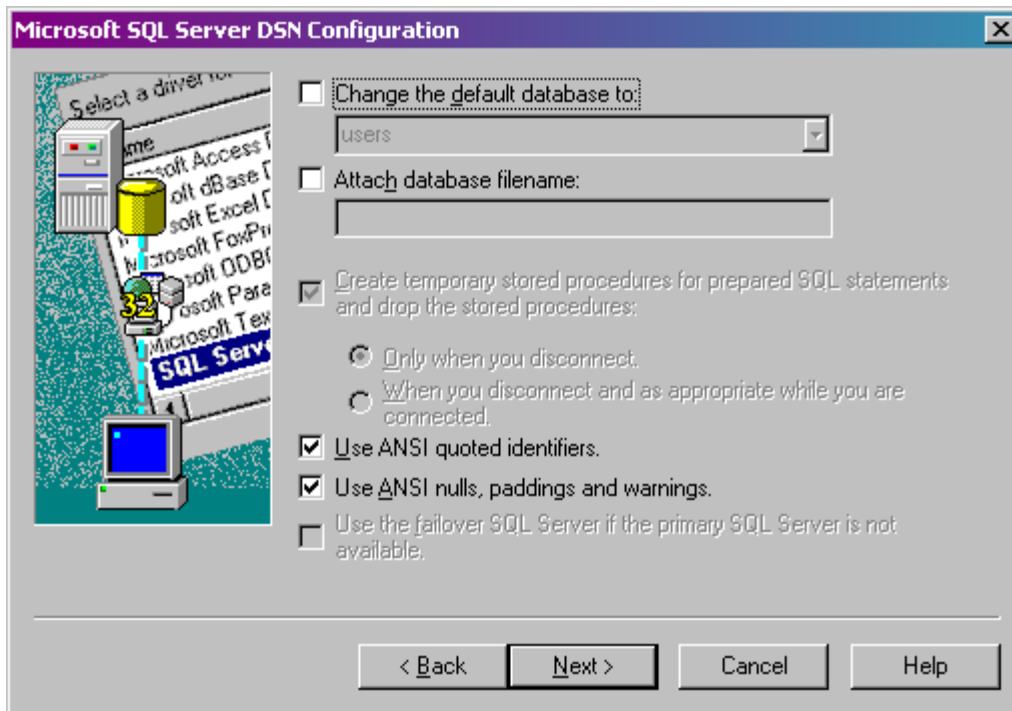
To change the network library used to communicate with SQL Server, click Client Configuration.

☒ Connect to SQL Server to obtain default settings for the additional configuration options.

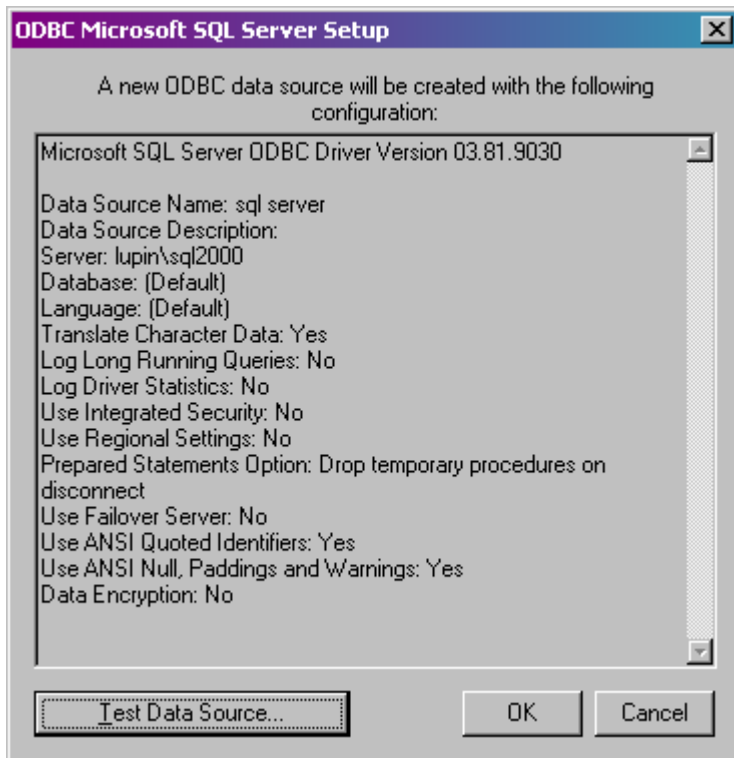
Login ID:

Password:

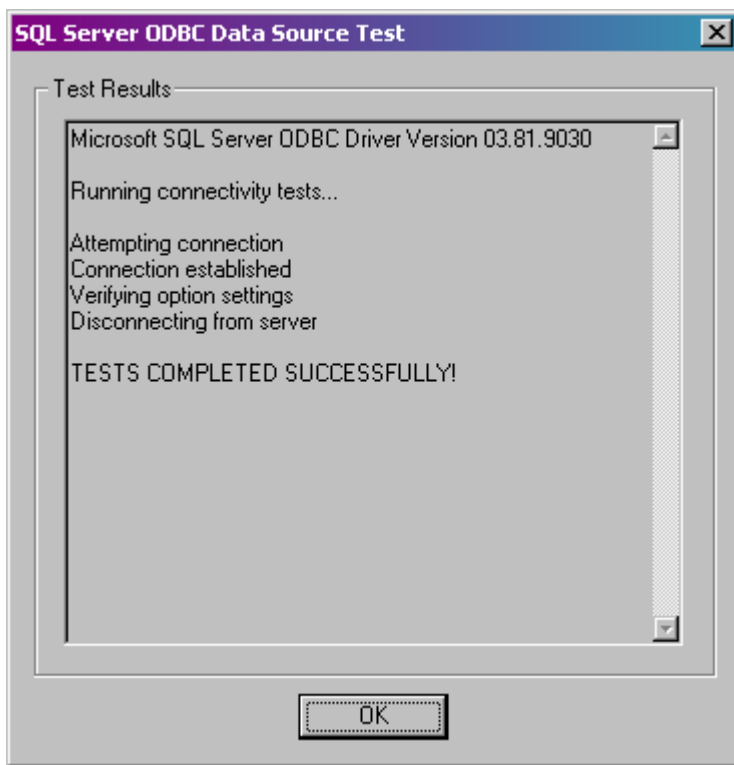
The next two screens allow for further server configurations, such as changing the default database, creating temporary stored procedures, changing the language of SQL server system messages, and so on.



After you click **Finish**, you see a summary window of the configurations you chose, and you can try a test connection to verify that the configurations are valid.



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





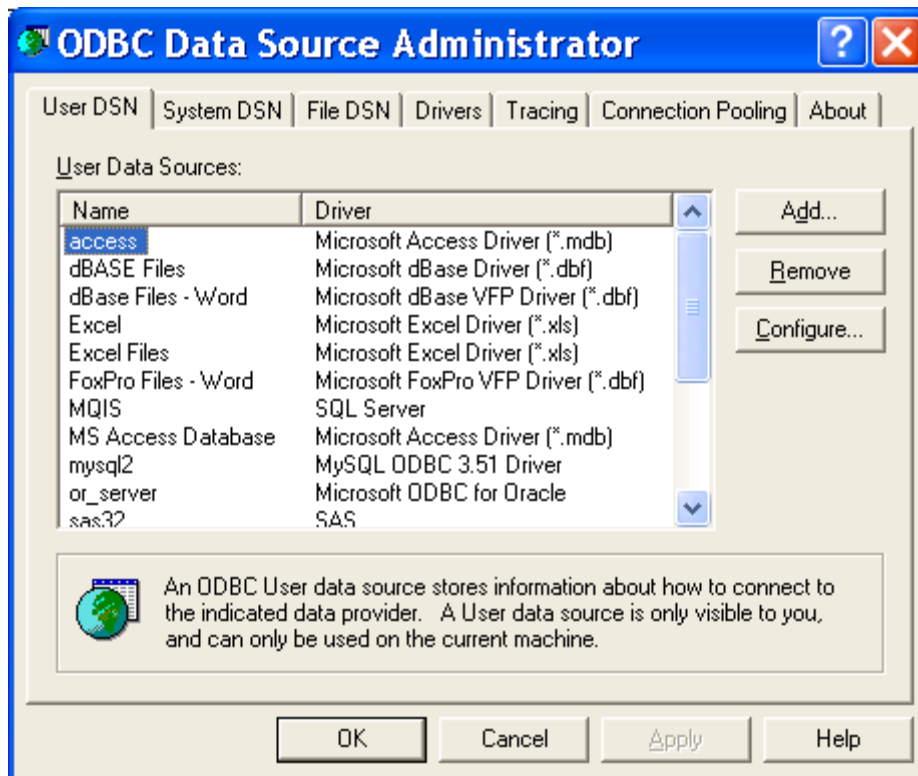
After the driver has been configured and the test connection is successful, then you can use a LIBNAME statement to create a library within SAS:

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

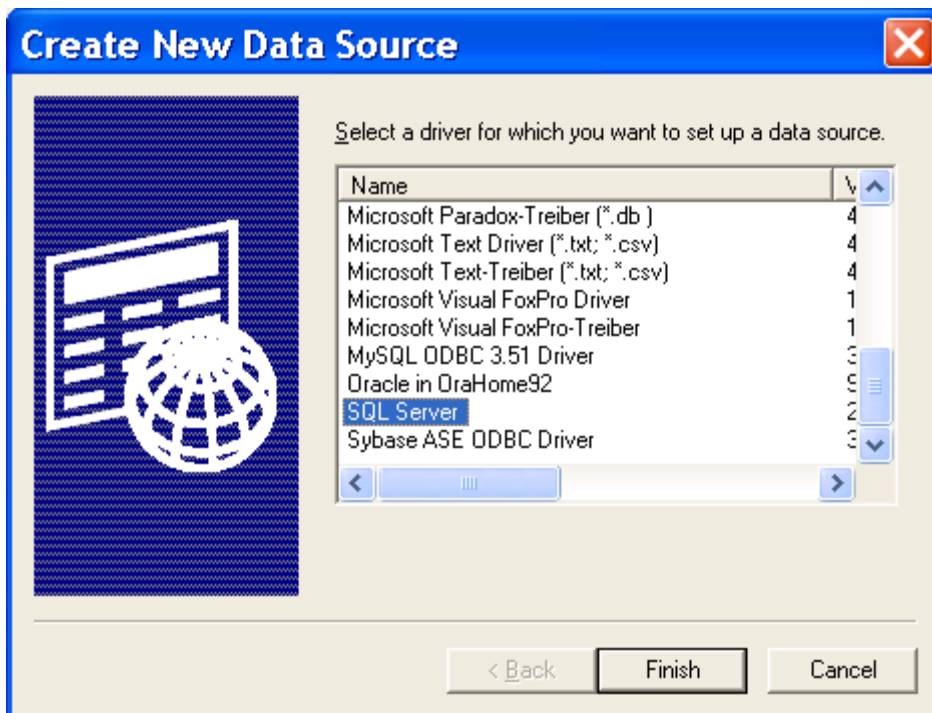
In the code above, 'sql server' is the name of the Data Source configured in the ODBC Administrator.

## Using NT Authentication

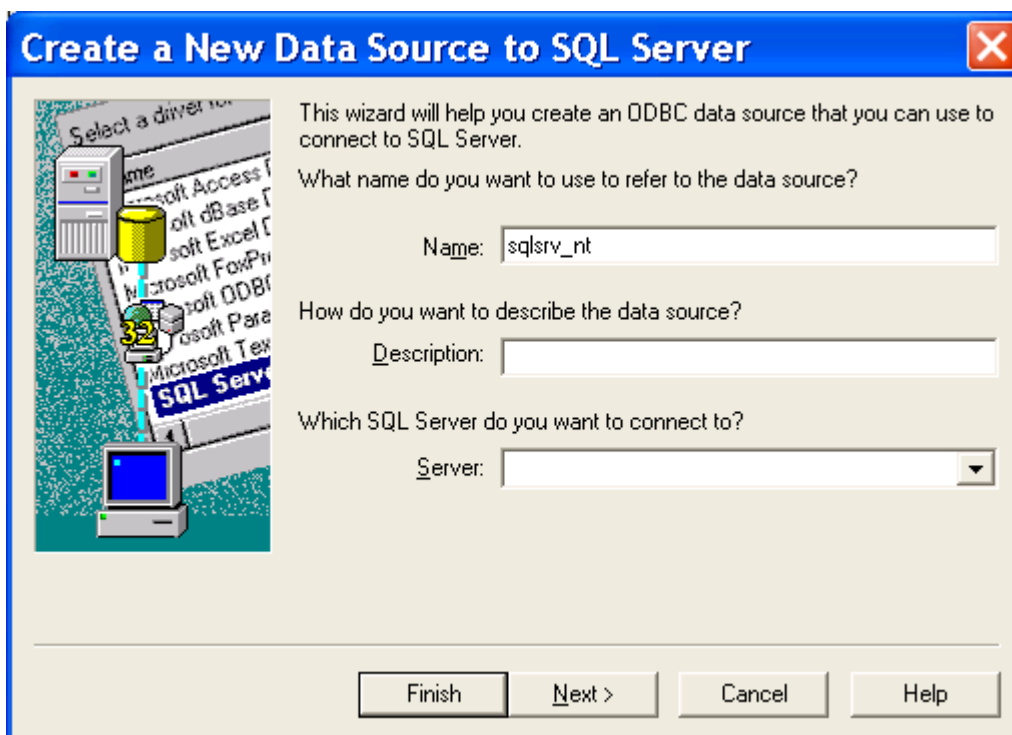
To set up an ODBC data source, select the **Start Menu**, click **Settings ► Control Panel**, and choose **Administrative Tools**. From there, choose **Data Sources (ODBC)**. This opens the ODBC Data Source Administrator dialog box.



Click the **User DSN** tab or the **System DSN** tab and click **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 **Next**.



Choose NT authentication as shown below and click **Next**.

## Create a New Data Source to SQL Server



How should SQL Server verify the authenticity of the login ID?

- ☒ With Windows NT authentication using the network login ID.
- ☐ With SQL Server authentication using a login ID and password entered by the user.

To change the network library used to communicate with SQL Server, click Client Configuration.

[Client Configuration...](#)

☒ Connect to SQL Server to obtain default settings for the additional configuration options.


Login ID:

Password:

[< Back](#)
[Next >](#)
[Cancel](#)
[Help](#)

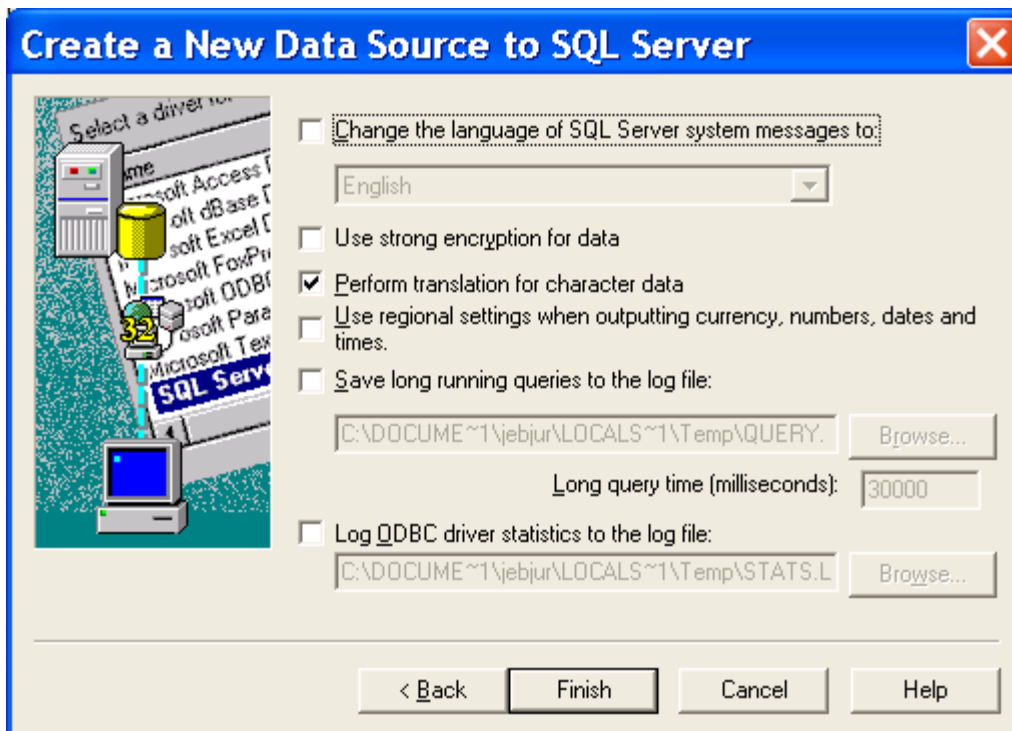
The next two screens allow for further server configurations, such as changing the default database, creating temporary stored procedures, changing the language of SQL server system messages, and so on.

## Create a New Data Source to SQL Server

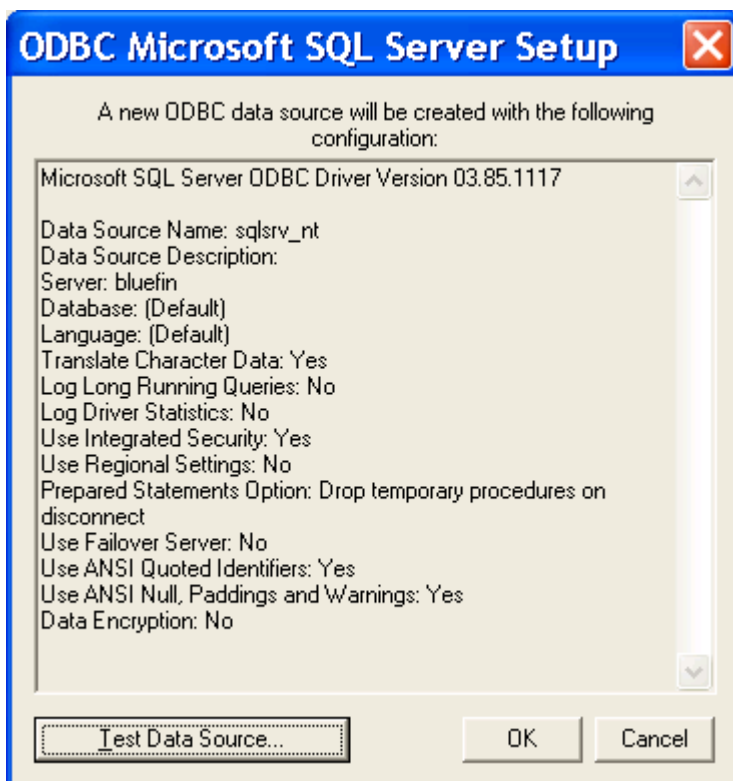


- ☐ Change the default database to:
- ☐ Attach database filename:
- ☒ Create temporary stored procedures for prepared SQL statements and drop the stored procedures:
  - ☒ Only when you disconnect.
  - ☐ When you disconnect and as appropriate while you are connected.
- ☒ Use ANSI quoted identifiers.
- ☒ Use ANSI nulls, paddings and warnings.
- ☐ Use the failover SQL Server if the primary SQL Server is not available.

[< Back](#)
[Next >](#)
[Cancel](#)
[Help](#)

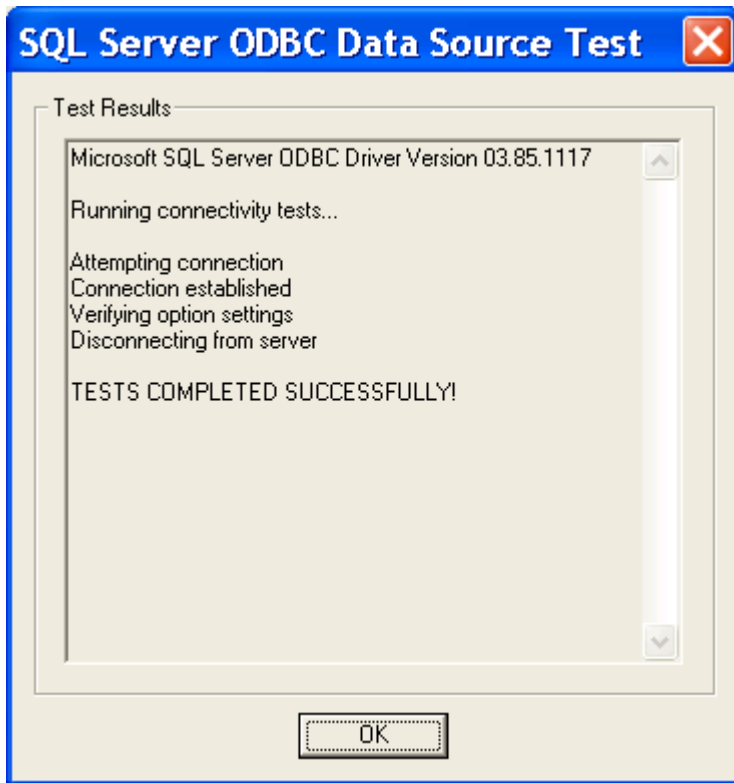


After you click **Finish**, you see a summary window of the configurations you chose, and you can try a test connection to verify that the configurations are valid.



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

Administrator.



After 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';
```

In the code above, 'sqlsrv\_nt' is the name of the Data Source configured in the ODBC Administrator.

## Prompted Connection

If you are not sure what values to add for the user ID, password, and data source, you can try connecting with a prompted connection. A *prompted connection* means that you are prompted to enter the above information instead of supplying it in the LIBNAME statement. Submit the following lines of code:

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

The Select Datasource window opens. If you created a user DSN or system DSN, you need to click the **Machine Data Source** tab. Choose the appropriate DSN, and click **Okay**. Enter your SQL Server login ID and password. After you have connected, several parameters will be written to the log window. Here is an example:

```
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 in the LIBNAME statement with a NOPROMPT= option as shown in the following example:

```
/* 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 into schemas, which are equivalent to database users or owners. In order to see particular tables in a defined library, you might need to add the SCHEMA= option to the LIBNAME statement. If no schema is specified, SAS searches the current user ID's schema by default.

For example:

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

If you are not sure which schema your tables are contained in, you can use one of the following methods to find it:

- Use the SAS Query Window: from the **Tools ► Query** menu.  
When the Query Window has loaded, go to **Tools ► Switch Access Mode ► ODBC**. Then, select your data source and respond to any prompts. When you are connected, you see a list of available tables from your odbc data source. The tables are two-level names such as dbo.table1. The first level (dbo) is the schema.
- Use PROC SQL pass-through method:  
This method creates a temporary data set with the list of available tables in the database. The TABLE\_SCHEM variable contains 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

After 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 example:

```
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 the PROC SQL pass-through with SAS/ACCESS to ODBC. The query looks similar to the following:

```
/* 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 OLE DB

With SAS/Access to OLE DB, you will utilize the SQL Server OLE DB data provider to work with your SQL Server database server and tables.

### SQL Server Authentication

With SAS/ACCESS to OLE DB, you do not have to configure the data provider. The LIBNAME statement looks similar to the following:

```
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 looks similar to the following:

```
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 are not sure which values to add for the user ID, password, and data source (=server name), then you can try connecting with a prompted connection. A prompted connection means that you are prompted to enter the above information instead of supplying it in the LIBNAME statement. Submit the following lines of code:

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

1. In the Data Link Properties pop-up window, select **Microsoft OLE DB Provider for SQL Server**.
2. Click **Next**.
3. Enter the Data Source name (server).
4. Select the **Use a specific user name and password** radio button, and enter the appropriate user name and

password.

5. Enter the name of a database on the server you wish to connect to (optional).
6. Select **Test Connection** and make sure it establishes a connection
7. Click **OK** to exit the pop-up window. 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 use an unprompted connection in the future, once you complete the steps above to establish a prompted connection to the SQL Server database, you would use the information written to the log in your LIBNAME statement. Specifically, the %PUT statement writes 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, 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 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";

/* 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 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 might be needed in the LIBNAME statement as well. If you need to find a schema for a table with SAS/ACCESS to OLE DB, you can use the PROC SQL pass-through code below. This method creates a temporary data set with the list of available tables in the database. The TABLE\_SCHEMA variable contains the schema.

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

After you find the appropriate schema value, add it to the LIBNAME statement with the SCHEMA= option. The LIBNAME statement looks 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

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



For example:

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

data new;
set sql.table1;
run;

proc sql;
create table new as select * from sql.table1;
quit;
```

You can also use PROC SQL pass-through with SAS/ACCESS to OLE DB. The query looks similar to the following:

```
/* 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 Institute Inc. (2006). SAS/ACCESS 9.1 Interface to Relational Databases. Cary, NC. Available at [support.sas.com/documentation/onlinedoc/91pdf/index\\_913.html#access](http://support.sas.com/documentation/onlinedoc/91pdf/index_913.html#access).



To contact your local SAS office, please visit: [sas.com/offices](https://sas.com/offices)

---

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies. Copyright © 2014, SAS Institute Inc. All rights reserved.