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

Contents

Introduction 1

Deciding Which SAS/ACCESS® Product to Use 1

SAS/ACCESS Interface to ODBC 2

  SQL Server Authentication 2

  Using NT Authentication 7

  Prompted Connection 11

  Schemas 13

  Reading Data 13

SAS/ACCESS Interface to OLE DB 14

  SQL Server Authentication 14

  Using NT Authentication 14

  Prompted Connection 15

  Schemas 16

  Reading Data 16

Resources 17

Introduction

A benefit of working with the relational database SQL Server is that it does not require much configuration to start working with the data. You do not have to install database client software, 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 article describes 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: Either SAS/ACCESS® Interface to ODBC or SAS/ACCESS® Interface 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. Both connection methods are discussed below.

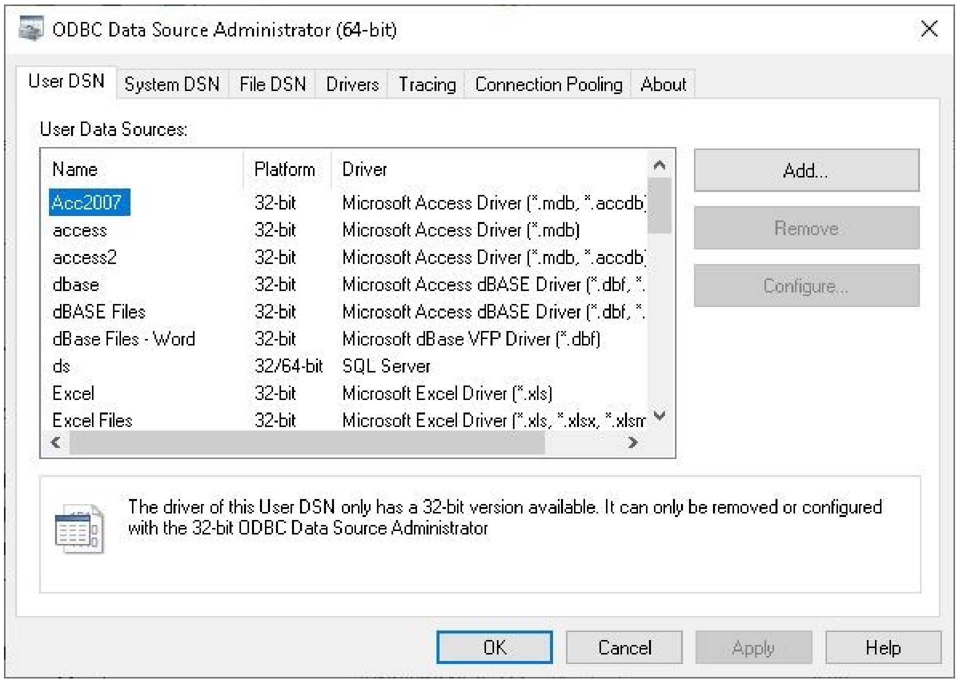
SAS/ACCESS Interface to ODBC

With SAS/ACCESS Interface to ODBC, you 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, complete these steps:

1. Click the **Start Menu**, and select **Control Panel ▶ System and Security ▶ Administrative Tools**.
2. Choose **ODBC Data Sources** (32bit or 64bit, depending on the bitness of SAS). This opens the ODBC Data Source Administrator dialog box. You can also access the **ODBC Data Source Administrator (64bit)** by double-clicking **ODBCAD32.exe** in **`C:\Windows\System32`**. The **ODBC Data Source Administrator (32bit)** is accessed by clicking **ODBCAD32.exe** in **`C:\Windows\SysWOW64`**.


ODBC Data Source Administrator (64-bit)

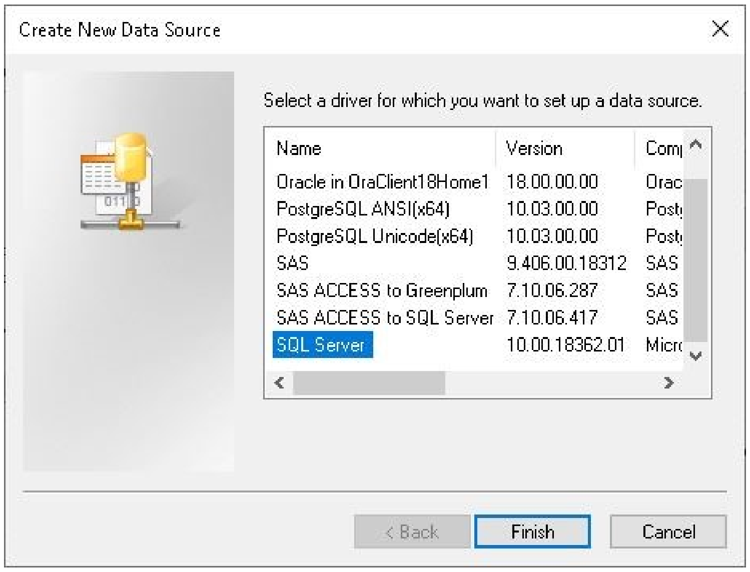
User DSN | System DSN | File DSN | Drivers | Tracing | Connection Pooling | About

User Data Sources:

Name	Platform	Driver
Acc2007	32-bit	Microsoft Access Driver (*.mdb, *.accdb)
access	32-bit	Microsoft Access Driver (*.mdb)
access2	32-bit	Microsoft Access Driver (*.mdb, *.accdb)
dbase	32-bit	Microsoft Access dBASE Driver (*.dbf, *.
dBASE Files	32-bit	Microsoft Access dBASE Driver (*.dbf, *.
dBase Files - Word	32-bit	Microsoft dBase VFP Driver (*.dbf)
ds	32/64-bit	SQL Server
Excel	32-bit	Microsoft Excel Driver (*.xls)
Excel Files	32-bit	Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm

The driver of this User DSN only has a 32-bit version available. It can only be removed or configured with the 32-bit ODBC Data Source Administrator

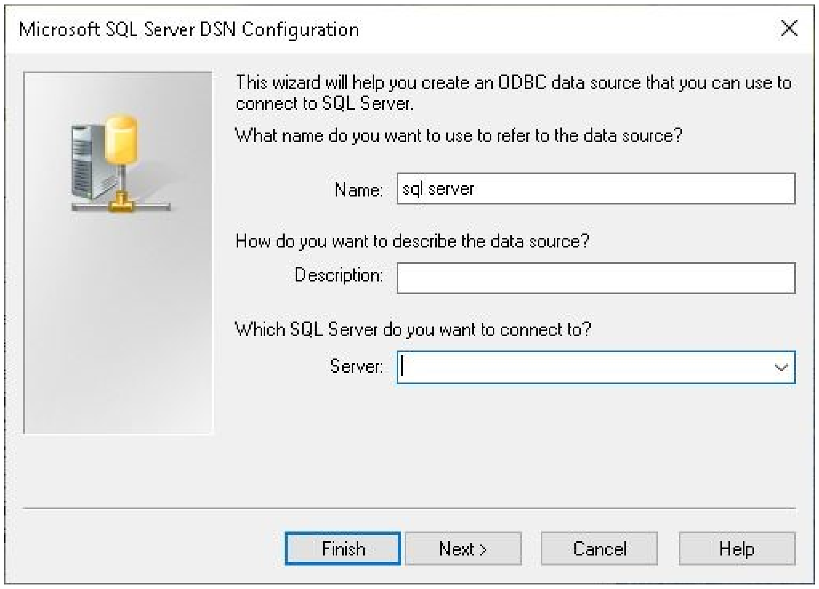
3. Click the **User DSN** tab or the **System DSN** tab and click **Add** to add a new data source.
4. Select the **SQL Server** driver and click **Finish**.


Create New Data Source

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

Name	Version	Comp
Oracle in OraClient18Home1	18.00.00.00	Orac
PostgreSQL ANSI(x64)	10.03.00.00	Post
PostgreSQL Unicode(x64)	10.03.00.00	Post
SAS	9.406.00.18312	SAS
SAS ACCESS to Greenplum	7.10.06.287	SAS
SAS ACCESS to SQL Server	7.10.06.417	SAS
SQL Server	10.00.18362.01	Micro

The next window enables 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: sql server

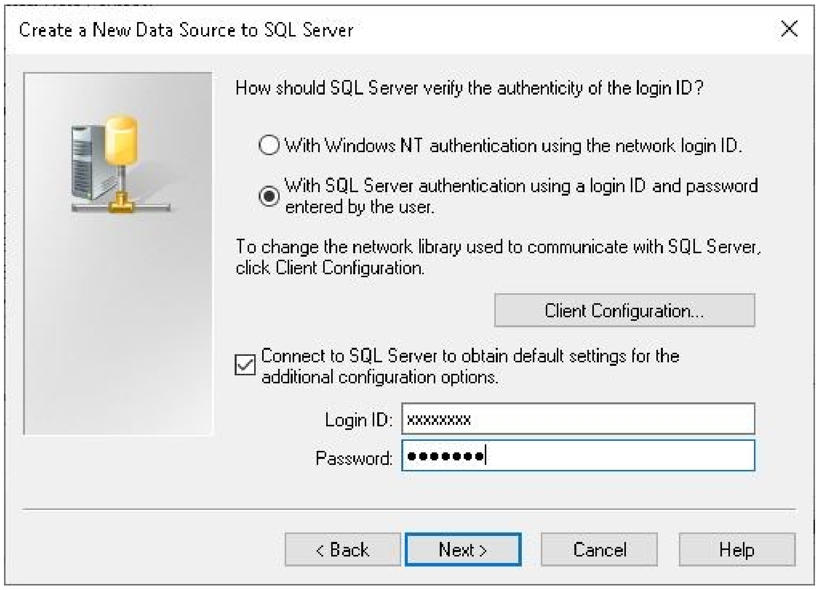
How do you want to describe the data source?

Description:

Which SQL Server do you want to connect to?

Server:

5. Choose SQL Server authentication as shown below and enter the SQL server login ID and password. 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.

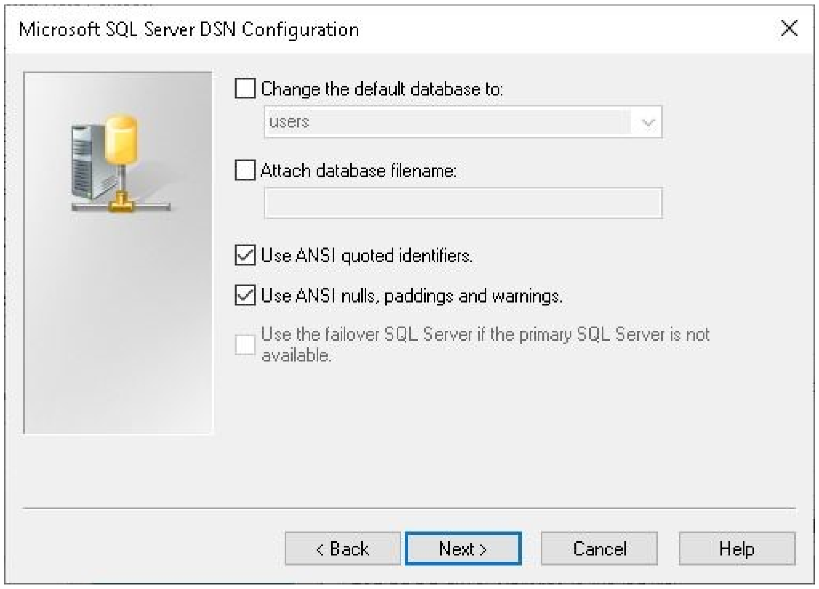
To change the network library used to communicate with SQL Server, click Client Configuration.

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

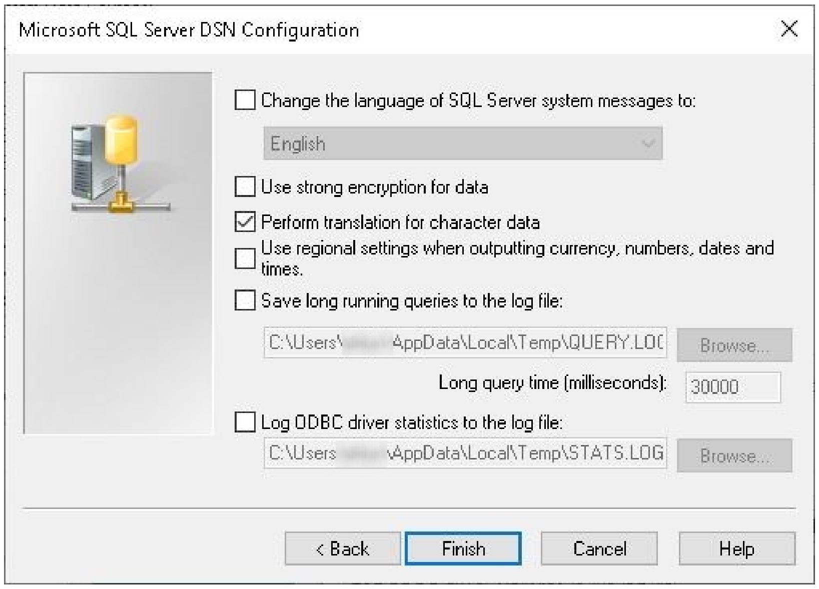
Login ID: xxxxxxxx

Password: •••••••

The next two screens enable further server configurations, such as changing the default database, changing the language of SQL server system messages, and so on.


Microsoft SQL Server DSN Configuration

- [ ] Change the default database to: users
- [ ] Attach database filename:
- [x] Use ANSI quoted identifiers.
- [x] Use ANSI nulls, paddings and warnings.
- [ ] Use the failover SQL Server if the primary SQL Server is not available.

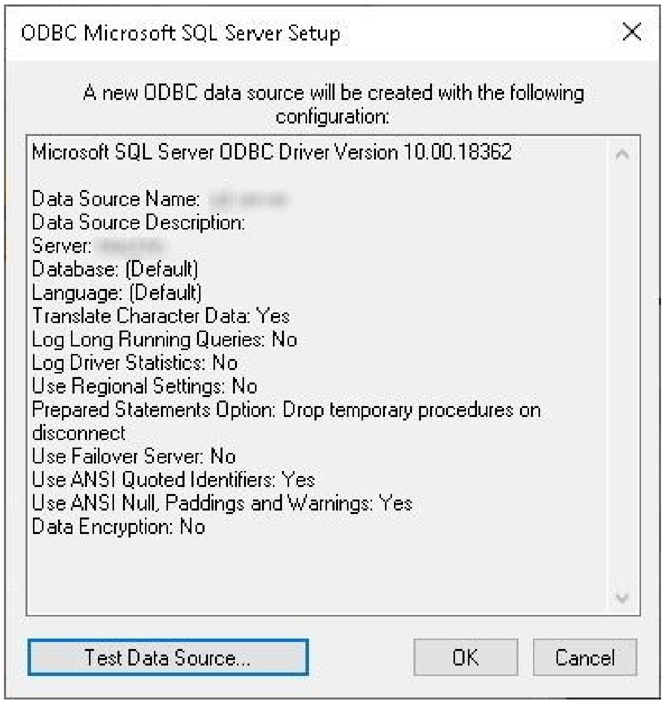

Microsoft SQL Server DSN Configuration

- [ ] Change the language of SQL Server system messages to: English
- [ ] Use strong encryption for data
- [x] Perform translation for character data
- [ ] Use regional settings when outputting currency, numbers, dates and times.
- [ ] Save long running queries to the log file: C:\Users\ AppData\Local\Temp\QUERY.LOC

Long query time (milliseconds): 30000

- [ ] Log ODBC driver statistics to the log file: C:\Users \AppData\Local\Temp\STATS.LOG

6. 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.



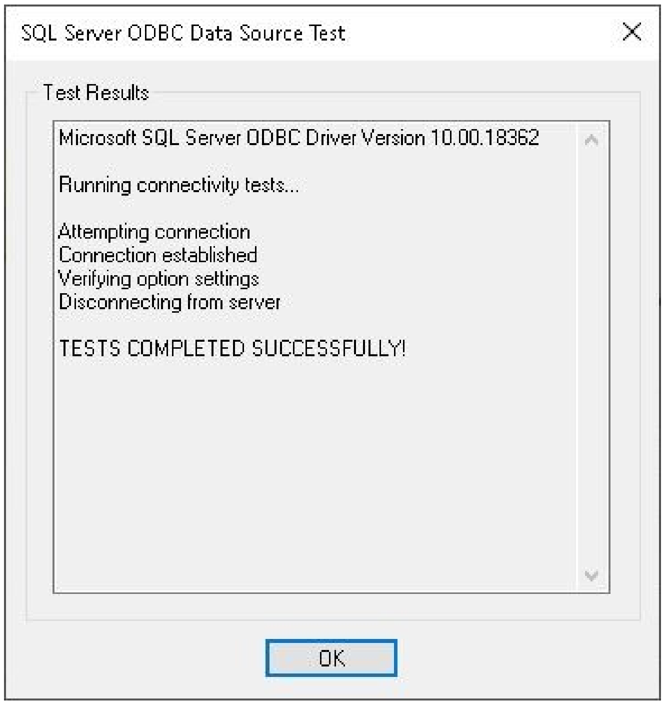
A new ODBC data source will be created with the following configuration:

```
Microsoft SQL Server ODBC Driver Version 10.00.18362

Data Source Name: [redacted]
Data Source Description:
Server: [redacted]
Database: (Default)
Language: (Default)
Translate Character Data: Yes
Log Long Running Queries: No
Log Driver Statistics: 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
```

If everything is set up properly, the test connection will be successful.

7. Click **OK** to exit the SQL Server setup and Administrator.



```
Microsoft SQL Server ODBC Driver Version 10.00.18362

Running connectivity tests...

Attempting connection
Connection established
Verifying option settings
Disconnecting from server

TESTS COMPLETED SUCCESSFULLY!
```

8. 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:

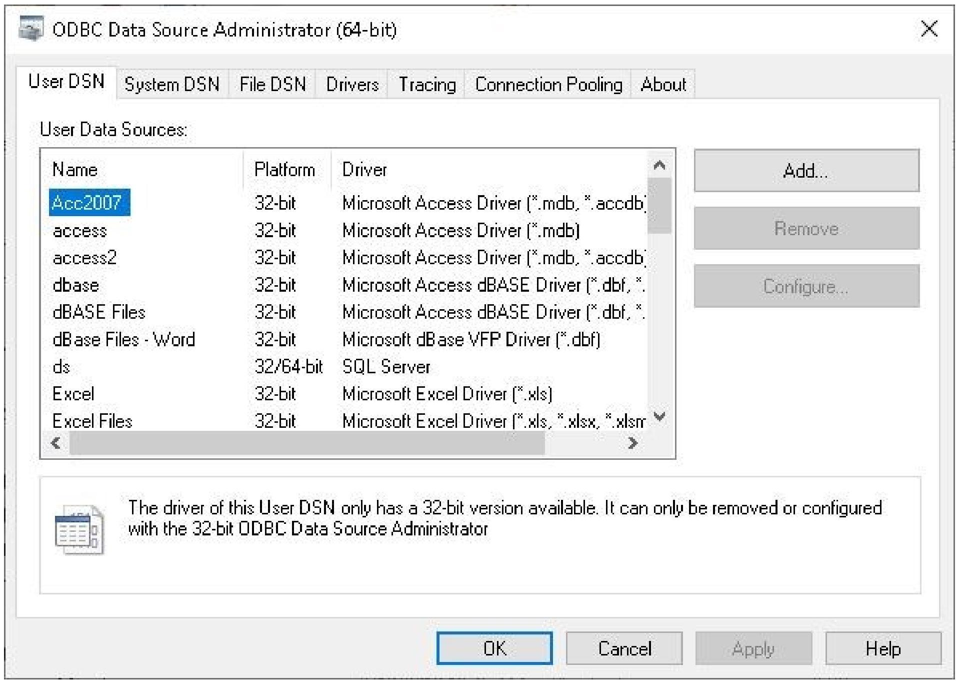
```
LIBNAME SQLLIB ODBC DSN='sql server' user=user pw=xxxx;
```

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, complete these steps:

1. Click the **Start Menu**, and select **Control Panel** ▶ **System and Security** ▶ **Administrative Tools**.
2. Choose **ODBC Data Sources** (32bit or 64bit, depending on the bitness of SAS). This opens the ODBC Data Source Administrator dialog box. You can also access the **ODBC Data Source Administrator (64bit)** by double-clicking **ODBCAD32.exe** in **`C:\Windows\System32`**. The **ODBC Data Source Administrator (32bit)** is accessed by clicking **ODBCAD32.exe** in **`C:\Windows\SysWOW64`**.

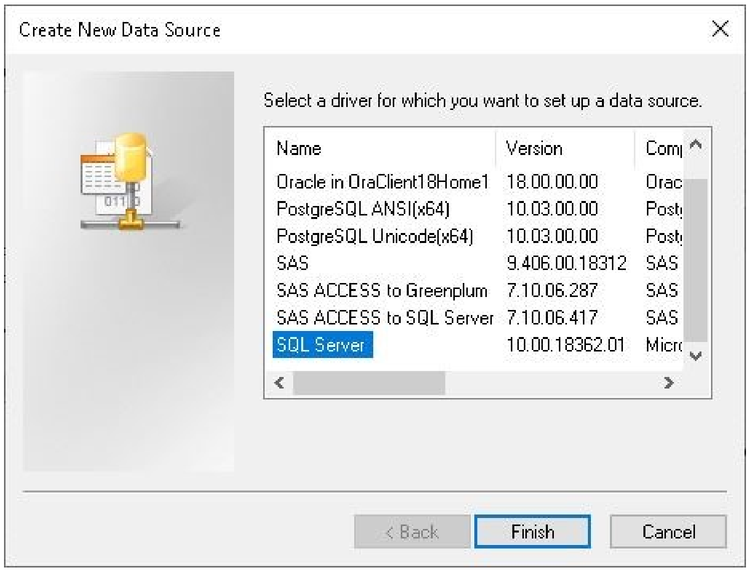


User Data Sources:

Name	Platform	Driver
Acc2007	32-bit	Microsoft Access Driver (*.mdb, *.accdb)
access	32-bit	Microsoft Access Driver (*.mdb)
access2	32-bit	Microsoft Access Driver (*.mdb, *.accdb)
dbase	32-bit	Microsoft Access dBASE Driver (*.dbf, *.
dBASE Files	32-bit	Microsoft Access dBASE Driver (*.dbf, *.
dBase Files - Word	32-bit	Microsoft dBase VFP Driver (*.dbf)
ds	32/64-bit	SQL Server
Excel	32-bit	Microsoft Excel Driver (*.xls)
Excel Files	32-bit	Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm

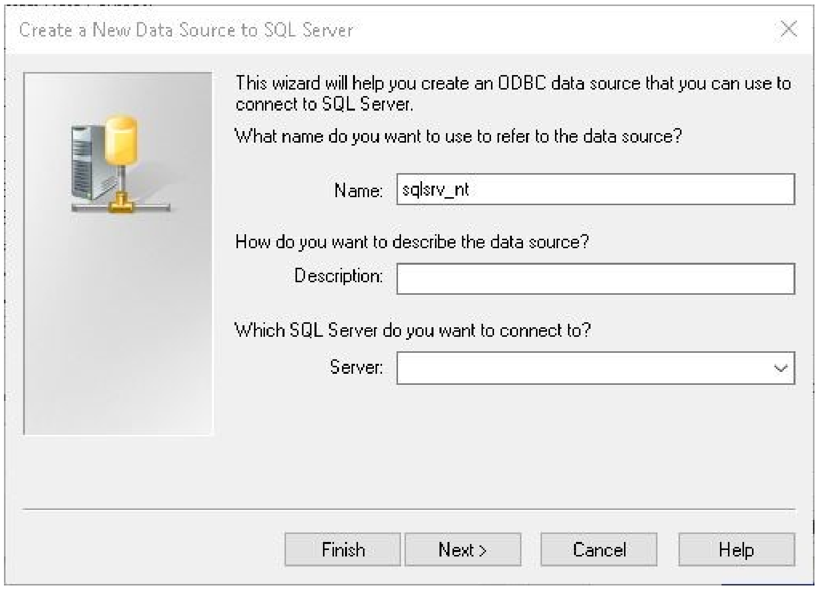
The driver of this User DSN only has a 32-bit version available. It can only be removed or configured with the 32-bit ODBC Data Source Administrator

3. 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**.



Name	Version	Com
Oracle in OraClient18Home1	18.00.00.00	Orac
PostgreSQL ANSI(x64)	10.03.00.00	Post
PostgreSQL Unicode(x64)	10.03.00.00	Post
SAS	9.406.00.18312	SAS
SAS ACCESS to Greenplum	7.10.06.287	SAS
SAS ACCESS to SQL Server	7.10.06.417	SAS
SQL Server	10.00.18362.01	Micr

The next window enables you to enter a name for the data source, an optional description, and the server you want to connect to. Click **Next**.



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: sqlsrv_nt

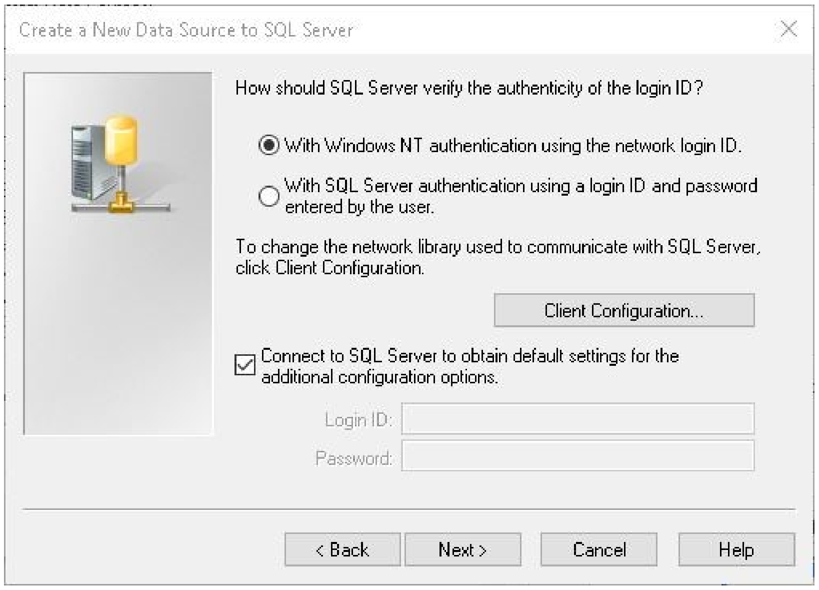
How do you want to describe the data source?

Description:

Which SQL Server do you want to connect to?

Server:

4. Select 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.

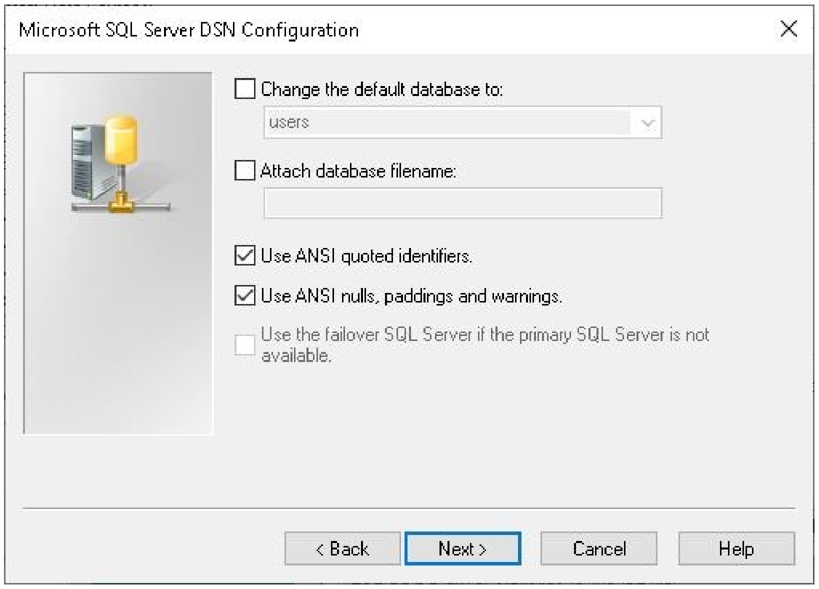
To change the network library used to communicate with SQL Server, click Client Configuration.

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

Login ID:

Password:

The next two screens enable further server configurations, such as changing the default database, changing the language of SQL server system messages, and so on.



Microsoft SQL Server DSN Configuration

☐ Change the default database to:

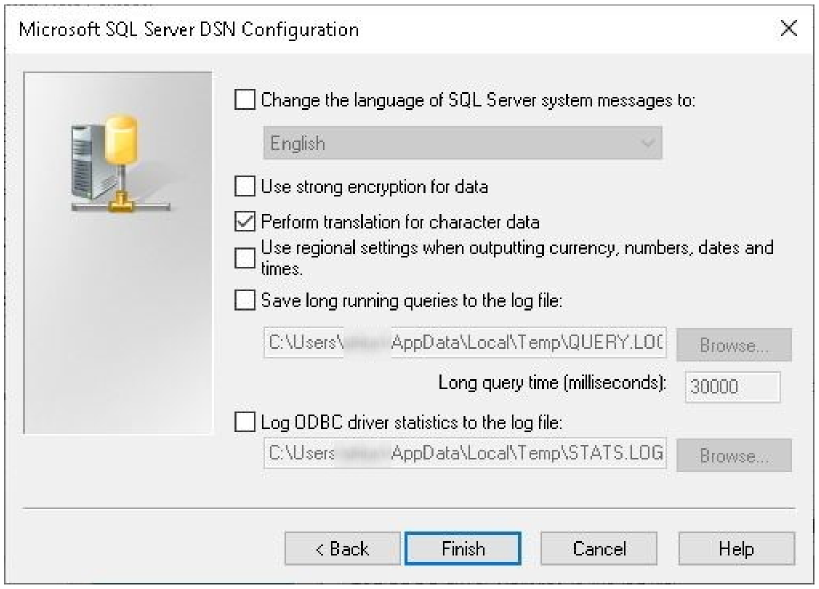
users

☐ Attach database filename:

☑ Use ANSI quoted identifiers.

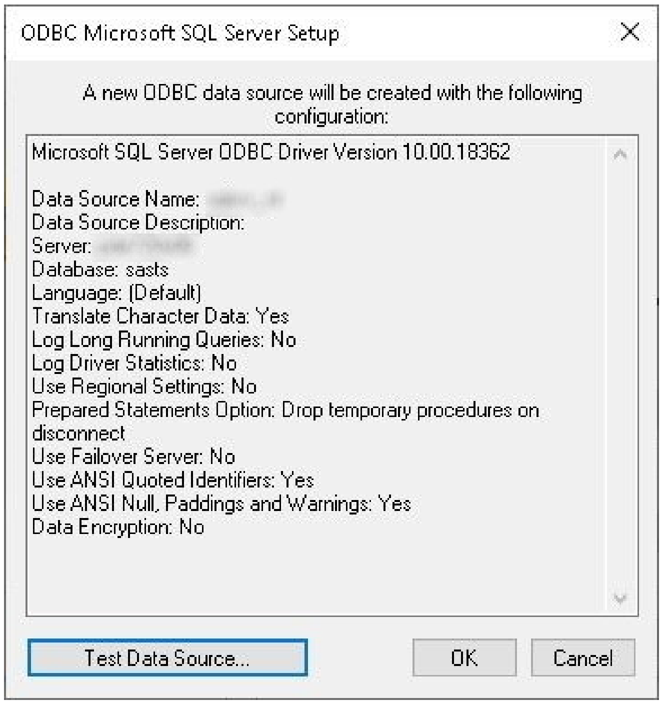
☑ Use ANSI nulls, paddings and warnings.

☐ Use the failover SQL Server if the primary SQL Server is not available.


Microsoft SQL Server DSN Configuration

- [ ] Change the language of SQL Server system messages to: English
- [ ] Use strong encryption for data
- [x] Perform translation for character data
- [ ] Use regional settings when outputting currency, numbers, dates and times.
- [ ] Save long running queries to the log file: C:\Users\ AppData\Local\Temp\QUERY.LOG
  Long query time (milliseconds): 30000
- [ ] Log ODBC driver statistics to the log file: C:\Users AppData\Local\Temp\STATS.LOG

5. 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.

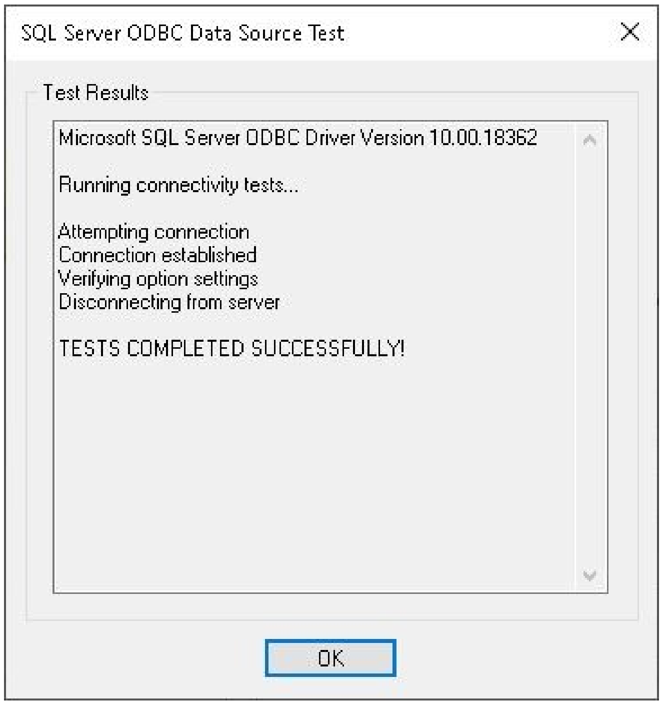

ODBC Microsoft SQL Server Setup

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

```
Microsoft SQL Server ODBC Driver Version 10.00.18362

Data Source Name:
Data Source Description:
Server:
Database: sasts
Language: (Default)
Translate Character Data: Yes
Log Long Running Queries: No
Log Driver Statistics: 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
```

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



```
Microsoft SQL Server ODBC Driver Version 10.00.18362

Running connectivity tests...

Attempting connection
Connection established
Verifying option settings
Disconnecting from server

TESTS COMPLETED SUCCESSFULLY!
```

7. 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.

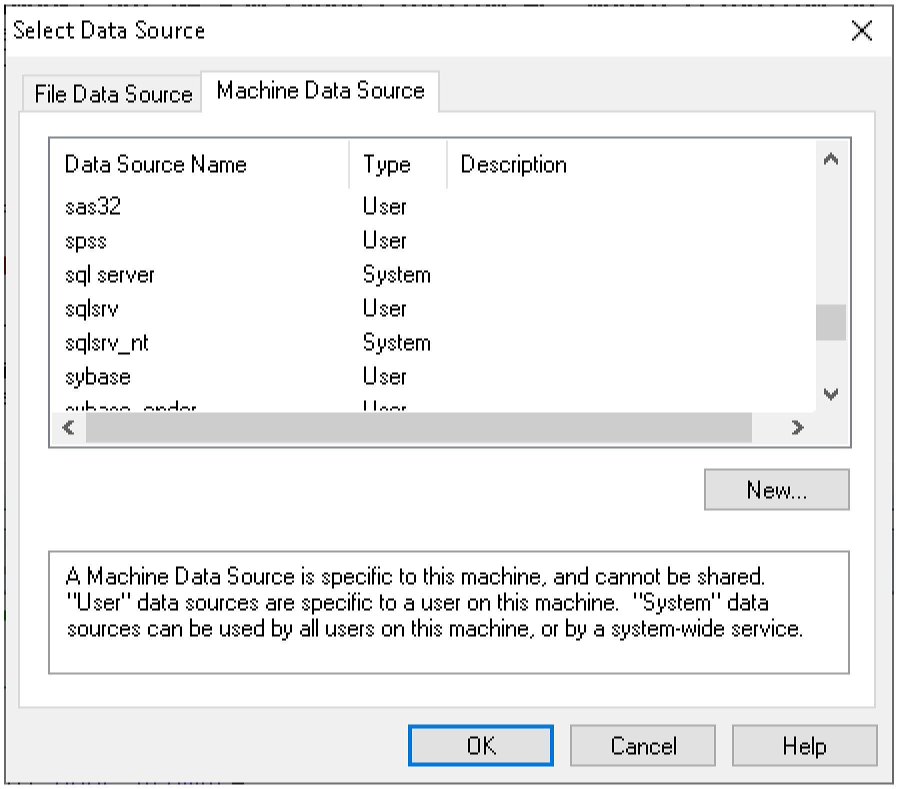
Prompted Connection

If you are not sure which 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.

1. Submit the following lines of code:

```
libname sql odbc prompt="";
%put %superq(sysdbmsg);
```

The **Select Data Source** window opens.



Data Source Name	Type	Description
sas32	User	
spss	User	
sql server	System	
sqlsrv	User	
sqlsrv_nt	System	
sybase	User	

A Machine Data Source is specific to this machine, and cannot be shared. "User" data sources are specific to a user on this machine. "System" data sources can be used by all users on this machine, or by a system-wide service.

2. If you created a user DSN or system DSN, you need to click the **Machine Data Source** tab.
3. Choose the appropriate DSN, and click **Okay**.
4. Enter your SQL Server login ID and password.
5. After you have connected, several parameters are written to the log window. Here is an example:

```
49   libname sql odbc prompt=XX;

NOTE: Libref SQL was successfully assigned as follows:
Engine:        ODBC
Physical Name: sqlsrv

50   %put %superq(sysdbmsg);

ODBC: DSN=sqlsrv;UID=user;PWD=xxxx;APP=SAS 9.4 for
Windows;WSID=ABC123
```

6. 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=user;PWD=xxxx;APP=SAS
9.4 for Windows;WSID=ABC123";


/* NT Authentication */
```

```
LIBNAME SQL ODBC noprompt=”dsn=sqlsrv_nt;WSID=ABC123;
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 SQLLIB ODBC DSN=sqlsrv user=user pw=xxxx 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 the 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 ;
  ```

Reading Data

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

For example:

```
  libname sqllib odbc dsn='sql server' user=user pw=xxxx schema=dbo;


data new;
  set sqllib.class;
  run;


proc sql;
  create table new as select * from sqllib.class;
  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 Interface to OLE DB

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

SQL Server Authentication

With SAS/ACCESS Interface 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=server_name";
```

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=dbase;
Data Source=server_name";
```

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);
```

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 that 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 use the information written to the log in your LIBNAME statement. Specifically, the %PUT statement writes the following to the log:

```
18   %put %superq(sysdbmsg);

OLEDB: Provider=SQLOLEDB.1;Password=xxxxx;Persist Security Info=True;User
ID=user;Initial Catalog=dbase;Data Source=server_name
```

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 sqllib oledb init_string="Provider=SQLOLEDB.1;Password=xxxxx;
Persist Security Info=True;User ID=user;Initial Catalog=dbase;Data
Source=server_name”;


/* NT Authentication */

libname sqllib oledb init_string="Provider=SQLOLEDB.1;Integrated
Security=SSPI;Persist Security Info=True;Initial Catalog=dbase;
Data Source=server_name”;
```

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 sqllib oledb  init_string="Provider=SQLOLEDB.1;Password=xxxxx;
   Persist Security Info=True;User ID=user;Initial Catalog=dbase;Data
   Source=server_name" schema=dbo;


   /* NT Authentication */

   libname sqllib oledb init_string="Provider=SQLOLEDB.1;Integrated
   Security=SSPI;Persist Security Info=True;Initial Catalog=dbase;
   Data Source=server_name" schema=dbo;
```

Reading Data

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

For example:

```
   libname sqllib oledb init_string="Provider=SQLOLEDB.1;Password=xxxxx;
   Persist Security Info=True;User ID=user;Initial Catalog=dbase;Data
   Source=server_name" schema=dbo;


 data new;
   set sqllib.class;
   run;

   proc sql;
   create table new as select * from sqllib.class;
   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=xxxx;
   Persist Security Info=True;User ID=user;Initial Catalog=dbase;Data
   Source=server_name" 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=dbase;
   Data Source=server_name” schema=dbo);
   select * from connection to oledb (select * from class);
   quit;
```

Resources

SAS Institute, Inc. 2021. *SAS/ACCESS® 9.4 Interface to Relational Databases: Ninth Edition*. Cary, NC. Available at go.documentation.sas.com/doc/en/pgmsascdc/9.4_3.5/acreldb/titlepage.htm