Paper SAS1789-2015

Step into the Cloud: Ways to Connect to Amazon Redshift with SAS/ACCESS®

James Ke Wang, SAS Research and Development (Beijing) Co., Ltd., and
Salman Maher, SAS Institute Inc.

ABSTRACT

Every day companies all over the world are moving their data into the cloud. While many options are available, much of this data will wind up in Amazon Redshift. As a SAS® user, you are probably wondering, "What is the best way to access this data using SAS?" This paper discusses the many ways that you can use SAS/ACCESS® to get to Amazon Redshift. We compare and contrast the various approaches to help you decide which is best for you. Topics that we discuss here are building a connection, choosing appropriate data types, and SQL functions.

INTRODUCTION

These days the cloud and cloud computing are becoming popular words in the computer industry worldwide. The new architecture brings better agility, flexibility, scalability, and cost savings than traditional services. Not only do more companies plan to adopt this new technology in their IT architecture, many businesses now depend on it. Database service is a core component that comprises a cloud system. The cloud also plays an important role as a sustaining system. Available from Amazon Web Services (AWS), Amazon Redshift provides users with a new option for data warehousing. SAS users can run statistics, analysis, and prediction work in the data warehouse. Indeed, you can access Amazon Redshift in many ways, such as through the PostgreSQL command utility or other GUI tools. This paper compares the three main Open Database Connectivity (ODBC) drivers—Progress DataDirect for Redshift driver, Amazon Redshift ODBC driver, and the PostgreSQL ODBC driver—using SAS/ACCESS® Interface to ODBC.

This paper covers these topics:

- differences when accessing Amazon Redshift using the ODBC driver from DataDirect, Amazon, or PostgreSQL
- main features of AWS, Amazon Redshift, and SAS/ACCESS Interface to ODBC
- some ways to build connections, plus read and write data from Amazon Redshift
- support for data types and SQL functions
- performance comparisons
- how to choose an appropriate ODBC driver to access Amazon Redshift

This paper includes several examples for these topics. These can help you choose an appropriate ODBC driver when accessing Amazon Redshift cluster through SAS software.

BACKGROUND

CLOUD COMPUTING PLATFORM

AWS is an IT infrastructure platform that originally offered services to businesses by way of web services. It is now commonly known as cloud computing and provides highly reliable, scalable, low-cost infrastructure.

Users might want to access the services in AWS from the SAS environment in the cloud. Doing this requires that you have an available AWS account. To learn how to create an AWS account, get help, and find more detailed information, go to http://aws.amazon.com.

One advantage of accessing AWS resource services from SAS software is that you can accomplish it through your existing installed SAS environment. However, when you plan to move large amounts of data between SAS software and AWS, keep in mind that this could lead to some performance risk, such as bandwidth limitations.

CLOUD DATA WAREHOUSE

Amazon Redshift is a fast, powerful, fully managed, petabyte-scale data warehouse service in AWS. It offers you fast query performance, and it makes it simple and cost-effective to efficiently analyze all your data using your existing business intelligence tools. One of the most significant advantages is that it has a similar SQL interface to PostgreSQL. Users can access the Amazon Redshift cluster through common PostgreSQL client tools, such as standard ODBC and JDBC drivers and the PSQL utility.

Unlike most other relational relational database management systems (RDBMSs), Amazon Redshift uses the column-oriented storage principle. It lets users handle analysis workloads on huge scale data sets. Another advantage is its query operations for data warehousing. It also uses a data distribution mode whereby stored rows can be distributed to node slices according to a distribution key that is defined for a table when data is loaded into the database service. An appropriate distribution key can allow the database to load data and execute queries in parallel and more efficiently.

You need to specify some SAS data set options in SAS software to define or configure one column as the distribution key when creating a table. One example in this paper demonstrates this point. To learn more about launching an Amazon Redshift cluster, go to http://docs.aws.amazon.com/redshift/latest/gsg/rs-gsg-launch-sample-cluster.html.

ODBC DRIVER

An ODBC driver uses Microsoft's ODBC interface, which allows applications to access data in the DBMS using SQL as a standard for accessing data. For a more detailed definition of this term, see http://en.wikipedia.org/wiki/Open_Database_Connectivity#Drivers.

Here are the three ODBC drivers we will cover in this paper:

- Progress DataDirect for ODBC for Amazon Redshift Wire Protocol by Progress Software Corporation
- Amazon Redshift ODBC by Amazon.com Inc.
- PostgreSQL ODBC by PostgreSQL.org

PROGRESS

The Progress DataDirect driver is an enterprise-level solution that is considered to be the best performing driver for writes to Redshift with an bulk-load option. For this paper we used the 7.1.4 ODBC driver that DataDirect provided to SAS. The official Progress website provides a trial version for download. For more detailed information about this driver, go to https://www.progress.com/products/datadirect-connect/odbc-drivers/data-sources/amazon-redshift.

The Amazon Redshift ODBC driver that we used for this paper was a pre-release version of the driver that is now available for download from http://docs.aws.amazon.com/redshift/latest/mgmt/configure-odbc-connection.html.

The PostgreSQL ODBC driver is usually known as psqlODBC. It is released under a GNU Lesser General Public License (LGPL). For details and download of the installation package based on to your operation system version, go to https://odbc.postgresql.org/.

SAS/ACCESS INTERFACES

We used SAS/ACCESS® Interface to ODBC to access Amazon Redshift with all three ODBC drivers, as mentioned in this paper.

SOFTWARE PREPARATION

A few software and configuration steps are needed to access Amazon Redshift. You must have an AWS account, an Amazon Redshift cluster instance, Base SAS®, SAS/ACCESS Interface to ODBC, and an available ODBC driver. If you run an application in an instance of Amazon EC2, you must launch it from your AWS console. If you want to load data into Amazon Redshift through the Amazon S3 service, you must also create a bucket through the Amazon S3 service.

Many online resources explain how to open and use an AWS account. You can buy a book or access online videos. This website provides a detailed guide about opening an AWS account: http://www.wikihow.com/Make-an-Amazon-Account.

To launch an Amazon Redshift cluster, you just need to log on from an AWS console and follow the instructions. For example, this website provides basic step-by-step instructions: http://docs.aws.amazon.com/redshift/latest/gsg/getting-started.html.

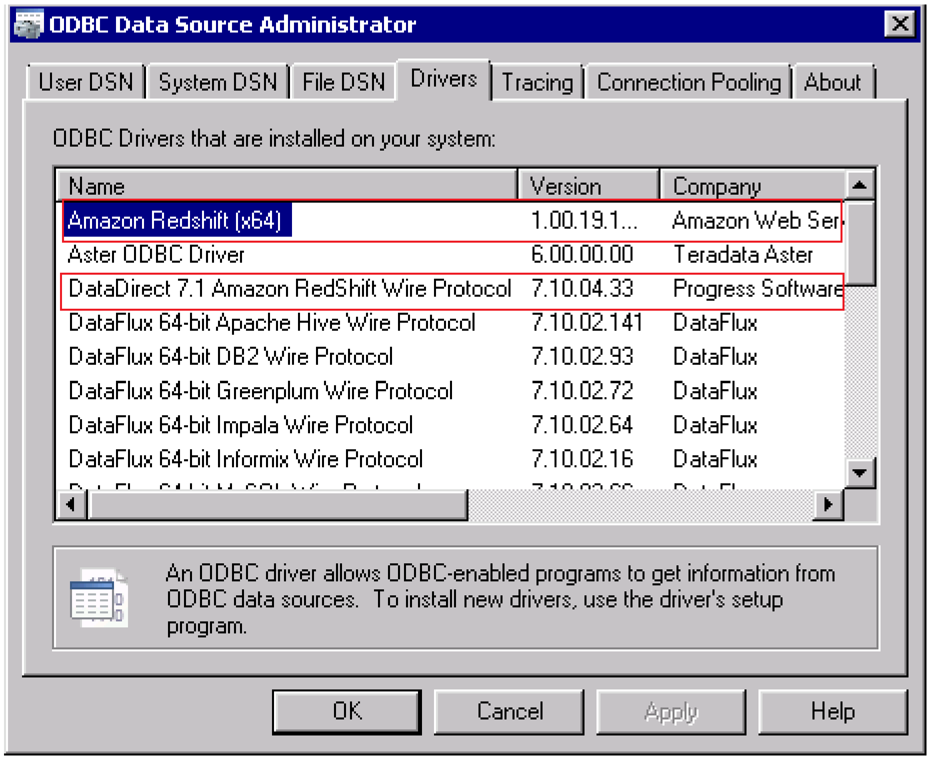
BUILDING CONNECTIONS

INSTALLING AN ODBC DRIVER

To connect to an Amazon Redshift cluster using SAS/ACCESS Interface to ODBC, you must provide a data source name (DSN) for the interface. Before doing this, however, be sure that the ODBC drivers are also already installed on your machine. When performing installation and configuration, there is no difference between the Redshift driver and other RDBMS-specific drivers. This paper focuses on installating on Windows.

Installation and configuration of these drivers on UNIX differs from that of Windows. In general you must do this by configuring or adjusting the relevant parameters in ODBC.INI and ODBCINST.INI. For more detail, search the internet or read the paper on this topic in the Reference section.

Display 1 shows an example of how to install ODBC drivers on a Windows x64 platform.


ODBC Data Source Administrator

ODBC Drivers that are installed on your system:

Name	Version	Company
Amazon Redshift (x64)	1.00.19.1...	Amazon Web Ser
Aster ODBC Driver	6.00.00.00	Teradata Aster
DataDirect 7.1 Amazon RedShift Wire Protocol	7.10.04.33	Progress Software
DataFlux 64-bit Apache Hive Wire Protocol	7.10.02.141	DataFlux
DataFlux 64-bit DB2 Wire Protocol	7.10.02.93	DataFlux
DataFlux 64-bit Greenplum Wire Protocol	7.10.02.72	DataFlux
DataFlux 64-bit Impala Wire Protocol	7.10.02.64	DataFlux
DataFlux 64-bit Informix Wire Protocol	7.10.02.16	DataFlux

An ODBC driver allows ODBC-enabled programs to get information from ODBC data sources. To install new drivers, use the driver's setup program.

Display 1. ODBC Driver Installation Example

Due to size limitations, the PostgreSQL driver isn't displayed in this window. Instead, tThe driver name that is displayed in the ODBC Data Source Administrator GUI shows as PostgreSQL Unicode(x64) or PostgreSQL ANSI(x64).

MAKE A LIBNAME CONNECTION

To use SAS/ACCESS Interface to ODBC to build a connection in Base SAS, create a data source name (DSN) and pass this in a LIBNAME statement.There is little difference when creating a DNS using the three ODBC drivers mentioned in this paper.

When creating the DSN, provide these details.

- server name: the server address of the Amazon Redshift cluster to be launched—for example, `examplecluster.myinstance.us-west-2.redshift.amazonaws.com`
- port: the service port number of the Amazon Redshift cluster to be launched—for example, the default (5439)
- database: the database name of the Amazon Redshift cluster to be launched
- user: a valid user name that has permission to access the Amazon Redshift cluster
- password: the user's authentication for a database

This example shows a common way to connect to an Amazon Redshift cluster using SAS/ACCESS to ODBC:

```
LIBNAME rhdd ODBC DATASRC='RH_DD_64BIT' USER='rsuser' pwd=xxxxxx;
```

Note: `RH_DD_64bit` is a DSN being configured using DataDirect ODBC driver.

Using the NOPROMPT= ODBC LIBNAME option lets you connect to Amazon Redshift without first defining a DSN--for example:

```
libname x odbc noprompt="Driver={Amazon Redshift (x64)};
Server=examplecluster.myinstance.us-west-2.redshift.amazonaws.com;
Database=testdb; UID=sasuser; PWD=insert_your_password; Port=5439" schema=public;
```

Note: Values that you set in the NOPROMPT= option are the same as those used to create the DSN.

FEATURES

DATA TYPE

For supporting Amazon Redshift, there is little difference in data type for the three drivers that are used in this paper. These drivers support the data types that Amazon Redshift supports when using SAS/ACCESS Interface to ODBC. For detailed documentation about data types that Amazon Redshift supports, go to http://docs.aws.amazon.com/redshift/latest/dg/c_Supported_data_types.html.

Nonetheless, there is still a subtle difference for data type support. One example is the BOOLEAN type: Both DataDirect and PostgreSQL drivers recognize BOOLEAN as a SQL_BIT type, while the Amazon Redshift driver handles it as a SQL_CHAR type. If you want to work with the Amazon Redshift driver to insert BOOLEAN data, use character string data for this type. This driver can accept TRUE or FALSE values. These two examples demonstrate the difference:

```
/* This example uses the Amazon Redshift driver. */
data rhamazon.james_datatype_amazon (dbtype=(cboolean='BOOLEAN'));
   cboolean = 'true';
   output;
run;

/* This example uses the DataDirect driver. */
data rhdd.james_datatype_datadirect (dbtype=(cboolean='BOOLEAN'));
   cboolean = 1;
   output;
run;
```

These subtle differences can make it difficult to switch among drivers based on the type of SAS jobs that you run. It would be best to pick and commit to a particular driver at the outset to avoid the possibility of rewriting your SAS jobs to accommodate ODBC driver differences.

NLS SUPPORT

Amazon Redshift supports multibyte characters through the VARCHAR type. As SAS users, you might want to read or write multibyte characters in the SAS system such as working under UTF-8 SAS session encoding. Currently, all three ODBC drivers can process multibyte data successfully using the SQL pass-through facility with SAS/ACCESS Interface to ODBC. These example use a DSN that is configured with the DataDirect driver to insert two Chinese characters using the SQL pass-through facility.

```
proc sql;
   CONNECT TO ODBC AS rslib (DATASRC = 'RH_DD_64bit' USER=james PWD=xxxxxx);
   EXECUTE (INSERT INTO james_nls_rhdd VALUES ('你好')) BY rslib;
   DISCONNECT FROM rslib;
quit;
```

If multibyte data output using SAS SQL is critical to your solution, it is recommended that you validate your usage case with the DataDirect driver before committing to a driver. We saw inconsistent results when we tested this with multiple languages. However, the Amazon Redshift and PostgreSQL drivers performed as expected—for example:

```
proc sql;
  create table rhamazon.james_nls_rhdd (txt VARCHAR(12));
  insert into rhamazon.james_nls_rhdd values ('你好');
quit;
```

Also, none of the three drivers work well with a DATA step to create a table and write data to Amazon Redshift. This is likely due to an existing issue with SAS/ACCESS Interface to ODBC. For example, this code fails when writing data:

```
 data rhdd.james_nls_rhdd_nlsdata;
   set work.nlsdata;
 run;
```

SQL FUNCTION SUPPORT

To process SQL functions, Amazon Redshift differs from most other traditional RDBMs. Redshift processes an SQL function on a leader node or distributes it to a compute node based on the type of SQL function. SAS/ACCESS provides a technology known as SQL pass-down. It maps part of SAS function to a DBMS function to maximize the amount of SQL that is being processed directly in the database to achieve higher performance.

When you use SAS/ACCESS Interface to ODBC with the drivers mentioned in this paper, there is little difference among them, primarily because these drivers comply with SQL industry standards.

This example lists several SAS functions that can't be passed down to Amazon Redshift directly using SAS/ACCESS Interface to ODBC.

- aggregate functions: STD, VAR
- string functions: STRIP, LOWCASE, UPCASE

You can customize the list of SAS functions that map to supported Redshift functions as needed by using the SQL_FUNCTIONS= option on the ODBC LIBNAME statement that you use to connect to Redshift. For details about this option, see the online SAS/ACCESS documentation. For detailed information about SQL functions that Amazon Redshift supports, see the online *Amazon Redshift Database Developer Guide*.

SQL OPERATION SUPPORT: UPDATE AND DELETE

Working with SAS/ACCESS Interface to ODBC, SAS users can send UPDATE or DELETE requests to a table that is stored in Amazon Redshift cluster. You can typically handle such operation by running an UPDATE SQL statement using the SQL procedure in the SQL pass-through facility. In addition, SAS users can update rows using the MERGE procedure.

Here is the difference in how SAS users can work with the ODBC drivers mentioned in this paper. With the PostgreSQL driver, you might see an error message similar to "`column ctid does not exist.`" This kind of error occurs because Amazon Redshift uses a columnar storage policy, which is different from that used by other traditional RDBMSs such as PostgreSQL and MySQL. Row-oriented RDBMSs typically use an additional column to store and mark the physical storage location of a row in a table. For example, the PostgreSQL RDBMS uses a column named CTID for this purpose. Also, the value that is stored in the column is used to track and locate each row when the PostgreSQL driver updates it. Similarly, the Amazon Redshift driver might also hit such error when you work with it. Both of these drivers are likely based on the same source-code architecture.

By contrast, the DataDirect driver provides a better result than the other two ODBC drivers for UPDATE and DELETE operations. You must set "UPDATE_SQL=NO" on the LIBNAME statement to force the engine to call the SQLSetPos ODBC API instead of generating an SQL query that Amazon Redshift doesn't support.

The code in this example demonstrates the UPDATE and DELETE operation:

```
libname rhdd odbc  datasrc='RH_DD_64bit'

   user=james pwd=Sasuser1 UPDATE_SQL=NO;

proc sql;

 update rhdd.james_upd_test set name='BJ' where no=3;

quit;
```

DATA READING AND WRITING

As an analytical database, Amazon Redshift features focus mainly on OLAP functions. Data movement—reading and writing—between SAS software and Amazon Redshift is a very basic requirement for some users. On one hand, SAS users have much experience with data analysis and processing in the SAS system and so prefer to use SAS in their tasks. On the other hand, users also want to leverage Amazon Redshift for large data storage and run some OLAP tasks. A common method in this case is to read raw data from a database into a SAS data set, process data in the SAS system, and then write the results to a database in a SAS data set.

SAS users can read and write data between an Amazon Redshift cluster and the SAS system using SAS/ACCESS Interface to ODBC with one of the three ODBC drivers in this paper inside or outside of AWS. Most SAS users take great care in performing such operations than with other more common operations. This paper uses the example below to show and verify performance on data transfer between Amazon Redshift and the SAS system. The example also compares the results when using the three ODBC drivers.

TEST SETTING

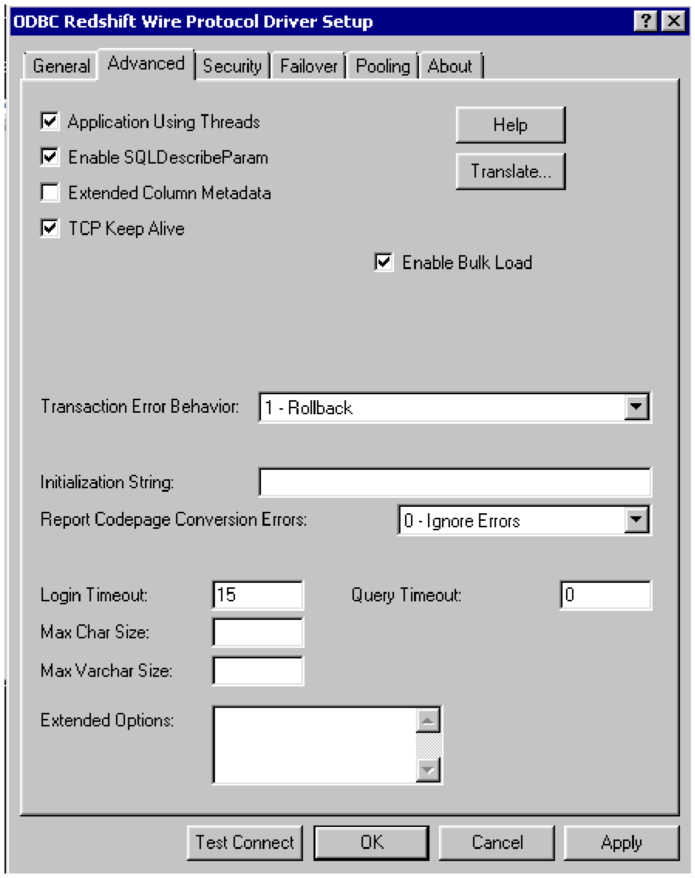
- **Software configuration**
  Windows 2008 R2 Standard with Service Pack 1
  SAS 9.4 TS Level 1M3 X64_S08R2
  Amazon Redshift ODBC driver(x64) 1.00.19.1021
  DataDirect Amazon Redshift Wire Protocol 7.10.33
  PostgreSQL Unicode(x64) 9.03.03

- **Hardware**
  CPU: INTEL Xeon E5-4640 0 @2.4GHz
  Memory: 16GB

Note: Because we did not have access to a SAS SID in AWS when preparing this paper, this example uses a virtual machine in an office instead of an EC2 instance in AWS.

- **Database**
  This example uses an Amazon Redshift cluster to be launched in Oregon.
  Cluster Name: sasredshift.clxtauywjji6.us-west-2.redshift.amazonaws.com

- **Data Source Setting**
  This example creates three data sources that correspond to each of the three ODBC drivers.


ODBC Redshift Wire Protocol Driver Setup

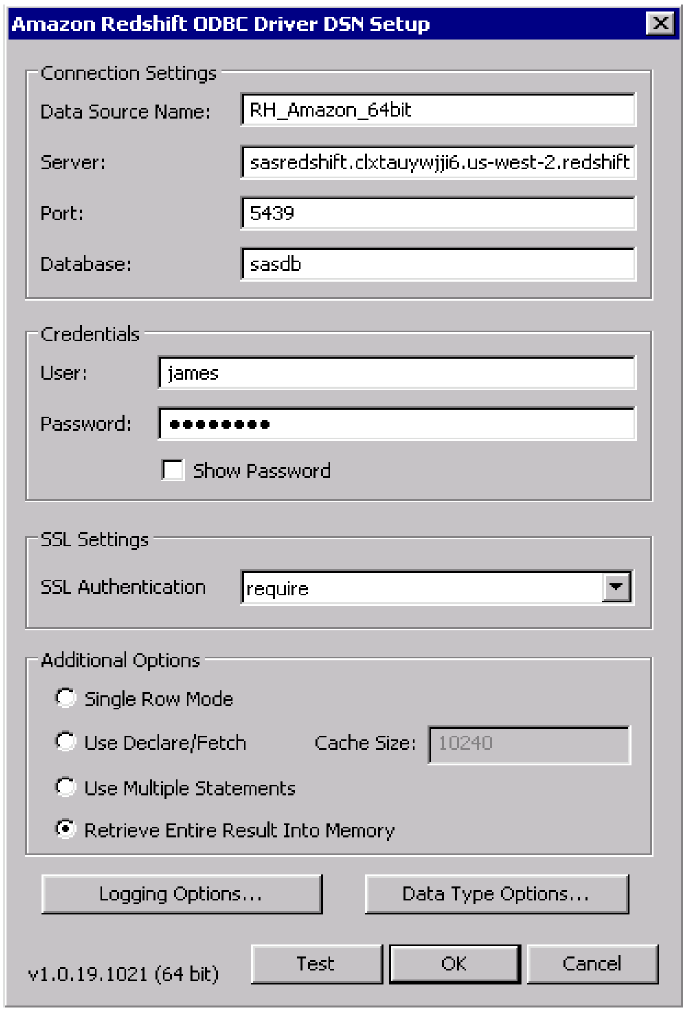
General | Advanced | Security | Failover | Pooling | About

☑ Application Using Threads
☑ Enable SQLDescribeParam
☐ Extended Column Metadata
☑ TCP Keep Alive
☑ Enable Bulk Load

Transaction Error Behavior: 1 - Rollback
Initialization String:
Report Codepage Conversion Errors: 0 - Ignore Errors
Login Timeout: 15
Query Timeout: 0
Max Char Size:
Max Varchar Size:
Extended Options:

Display 2. DataDirect ODBC Data Source Setup

Display 2 shows the configuration of a data source using the DataDirect ODBC driver. The **Enable Bulk Load** option is checked. Other options are the defaults.


Amazon Redshift ODBC Driver DSN Setup

Connection Settings

Data Source Name: RH_Amazon_64bit

Server: sasredshift.clxtauywjji6.us-west-2.redshift

Port: 5439

Database: sasdb

Credentials

User: james

Password: ●●●●●●●●●

☐ Show Password

SSL Settings

SSL Authentication: require

Additional Options

○ Single Row Mode

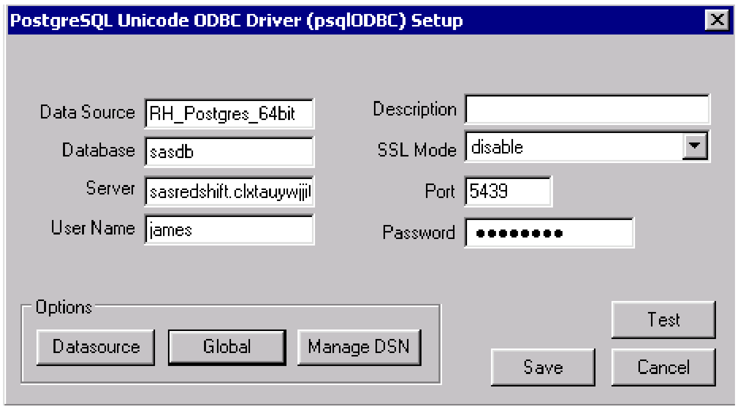
○ Use Declare/Fetch   Cache Size: 10240

○ Use Multiple Statements

● Retrieve Entire Result Into Memory

Display 3. Amazon Redshift ODBC Data Source Setup

Display 3 shows the configuration of a data source using the Amazon Redshift ODBC driver. The driver that we used while writing this paper was a pre-release version of the driver that is now available for download at http://docs.aws.amazon.com/redshift/latest/mgmt/configure-odbc-connection.html. When you configure this driver, you must enter a value for the **Cache Size** option when you create a data source because of a GUI bug in this dialog box. Other options show the default values.


PostgreSQL Unicode ODBC Driver (psqlODBC) Setup

Data Source: RH_Postgres_64bit

Database: sasdb

Server: sasredshift.clxtauywjji

User Name: james

Description:

SSL Mode: disable

Port: 5439

Password: ●●●●●●●●●

Display 4. PostgreSQL ODBC Data Source Setup

Display 4 shows the configuration of a data source using the PostgreSQL ODBC driver. You can also click **Datasource** to configure more options. We used the default settings for this example.

PERFORMANCE VERIFICATION

For this paper we used data from the cases below to verify read performance in the Amazon Redshift cluster.

In Case 1, the zipcode table that exists in Amazon Redshift contains 20 columns. The original table data comes from the zipcode data set in a SASHELP library in SAS that stores 10,000 rows. The READBUFF= option is set to 2000 to enable a large read buffer. This is just an example and does not mean that you must set the buffer size to 2000. Actually, you can set this buffer size to a larger value for improved performance, but this creates a larger memory footprint.

In Case 2, the code generates a data set that contains 1000 observations and four columns with a character string type. The case writes all rows out to a table in Redshift from the data set using APPEND procedure. The size of the INSERTBUFFER data set option is set to 4000. A large buffer size setting can improve I/O performance over a small size when you output data to Redshift, especially for DataDirect driver.

Case 1: READ Code Example

```
 data work.zipcode_rhdd;
   set rhdd.zipcode (readbuff=2000);
run;


data work.zipcode_rhamazon;
   set rhamazon.zipcode (readbuff=2000);
run;


data work.zipcode_rhpg;
   set rhpg.zipcode (readbuff=2000);
run;
```

Case 2: WRITE Code Example

```
data work.small_data;
   do i = 1 to 1000;
      w='this is a reasonably long string that will be used to add bulk to the data';
      x='this is a reasonably long string that will be used to add bulk to the data';
      y='this is a reasonably long string that will be used to add bulk to the data';
      z='this is a reasonably long string that will be used to add bulk to the data';
    output;
end;
run;
```

```
proc append base=rhdd.small_data (insertbuff=4000)
                data=work.small_data;
run;
```

TEST RESULT

Table 1 shows verification results for Case 1. It addresses read performance using SAS/ACCESS Interface to ODBC. Table 2 addresses write performance. Neither the Amazon Redshift and PostgreSQL drivers provide support for bulk-loading in DSN, so the relevant data isn't shown.

Driver Name	With Bulk Load	Without Bulk Load
Data Direct ODBC Driver	4.84 seconds	5.13 seconds
Amazon Redshift ODBC Driver	N/A	5.17 seconds
PostgreSQL ODBC Driver	N/A	3.49 seconds

Table 1. Read Performance Result for Case 1

Driver Name	With Bulk Load	Without Bulk Load
Data Direct ODBC Driver	3.75 seconds	1:40.70
Amazon Redshift ODBC Driver	N/A	1:39.93
PostgreSQL ODBC Driver	N/A	1:46.75

Table 2. Write Performance Result for Case 2

UPLOADING DATA

Uploading of data is a common and necessary operation when dealing with big data and performing analysis on it. This is particularly true for SAS users who might want to load a large amount of data into Amazon Redshift from the SAS system. An example would be loading a million observations into an existing table in Amazon Redshift from a SAS work library on a local machine.

Many aspects influence the final result of uploading data such as time, costs, performance, data, and compatibility.

Performance, final time, and costs are typically key factors for most users, especially when uploading a large amount of data into this kind of database from a cloud storage environment.

This paper provides a method to let SAS users upload data to Amazon Redshift. This method doesn't necessarily have a relationship with SAS/ACCESS Interface to ODBC and the three ODBC drivers mentioned earlier.

LOADING METHOD

Amazon Redshift recommends that the COPY command be submitted to load data into Amazon Redshift from Amazon S3, Amazon DynamoDB Table, remote hosts, and several other Amazon online services using a DBMS client that connects to the Amazon Redshift cluster. This paper focuses on using the Amazon S3 in temporary file storage.

Here are the main steps: First, export data to a CVS format file using the EXPORT procedure and store the file in local disk. Next, upload the file into your buckets in Amazon S3. You might need to add those files into S3 using the Amazon S3 console. Last, execute a COPY command for the actual loading. The

COPY command can be submitted using the SQL pass-through facility or another SQL client such as the PSQL utility.

The example below shows the steps used in this method:

Step 1. Export data, store and update the file

```
proc export data=sashelp.zipcode (obs=40000)
                      outfile='C:\zipcode.csv'
                      dbms=csv
                      replace;
                      putnames=no;
run;
```

Step 2. Upload “zipcode.csv” to Amazon S3

Step 3. Execute a COPY command

```
proc sql;
connect to odbc as con_redshift (datasrc='RH_DD_64bit'
                                        user=james
                                        pwd=Sasuser1);
execute (copy zipcode from 's3://mybucket/zipcode.csv'
           credentials 'aws_access_key_id=xxxx;
                        aws_secret_access_key=xxxxxx';)
           by con_redshift;
disconnect from con_redshift;
quit;
```

CONCLUSION

Currently, SAS does not provide a native SAS/ACCESS interface product that is specifically for Amazon Redshift. Even so, for most SAS users, SAS/ACCESS Interface to ODBC is one appropriate way to access Amazon Redshift. You can also choose among several ODBC vendor-specific drivers to access Amazon Redshift. The PostgreSQL ODBC driver provides very basic access to Amazon Redshift. Amazon.com officially provides a specific ODBC driver that performs very similarly to the PostgreSQL driver. However, by comparison, the DataDirect driver provided by Progress provides a more robust and complete solution than either the PostgreSQL or Amazon drivers.

The DataDirect driver provides more advantages in write performance and in UPDATE and DELETE operations than the other drivers. If your solution requires a great number of writes to Redshift, using the DataDirect ODBC driver would be the best option.

REFERENCES

SAS/ACCESS® 9.4 for Relational Databases: Reference, Sixth Edition
http://support.sas.com/documentation/cdl/en/acreldb/67589/PDF/default/acreldb.pdf

Amazon Redshift Database Developer Guide
http://docs.aws.amazon.com/redshift/latest/dg/welcome.html

Bailey, Jeff. 2014. "An Insider's Guide to SAS/ACCESS® Interface to ODBC."
Proceedings of the SAS Global Forum 2014 Conference
http://support.sas.com/resources/papers/proceedings14/SAS039-2014.pdf

ACKNOWLEDGMENTS

The authors extend their heartfelt thanks and gratitude to these individuals:

Jeff Bailey, Cloudera Inc.
Junwu Ma, SAS Beijing R&D Ltd.
Pan She, SAS Beijing R&D Ltd.
Bill Oliver, SAS Institute
Chris Dehart, SAS Institute
Michael Ho, SAS Institute
Kathy Underwood, SAS Institute
Kat Turk, SAS Institute

CONTACT INFORMATION

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

James Ke, Wang
14/F Motorola Plaza, No.1 Wang Jing East Road,
Chao Yang District, Beijing, China 100102
+86 10 83193355
James.wang@sas.com
http://www.sas.com

Salman Maher
100 SAS Campus Drive,
Cary, NC 27513
SAS Institute Inc.
Salman.Maher@sas.com
http://www.sas.com