Paper 2357-2018

External Databases: Tools for the SAS® Administrator

Mathieu Gaouette, Prospective MG inc.

ABSTRACT

The SAS Management console administration tool allows the configuration of several external data sources for end users. The literature often diminish the complexity and importance of the configuration aspect of external databases making this task somehow ambiguous.

Most administrators find themselves overwhelmed by the number of options available to them when configuring these sources. As it is possible to fly through the configuration process by choosing the default values, the resulting database access can easily be flimsy or have sub-par performance.

This paper provides the administrator with a complete set of tools that will help them define from scratch an external database. It will also help optimize the available options (standards and advanced) in order to make the most out of that connection.

INTRODUCTION

Experience showed me that connecting SAS to external databases is a struggle in most cases for administrators. Although deep database knowledge is not needed for the SAS administrator, it is a very good strength. As a consequence, many administrators find themselves powerless when encountering database connection or performance issues.

Most of the challenges of maintaining SAS connections to external databases can be avoided by configuring correctly the connections. Most people will assume that a working connection is a good connection and there lies the problem. When testing a newly connected database, the administrator is not simulating a realistic everyday life with SAS users connecting simultaneously. Moreover, the administrator will generally test the connection with a basic query and not one that reflects the typical tasks of the SAS users.

This paper will cover the process of connecting an external database from start to finish. The goal is to explain the purpose of the common options between different databases and to point out the important parameters that should be tweaked. With this, the administrator will be able to connect and troubleshoot connections to external databases and also work with database administrators in order to implement configuration options to optimize performance.

CONFIGURATION LAYERS

A well-defined library to an external database is based on three levels of configurations. The first one is the OS level. In order for SAS to be able to access an external database, it needs proper configuration and database information to be configured. This layer includes the installation of the appropriated connecting drivers (such as DB2, Oracle to name just two). It also includes the registration of the actual database instance. Every database has its own way of mapping that information. For example, DB2 uses catalogs and Oracle uses tnsnames. Any network preparation such as firewall ports being opened should be taken care of here as well. Once completed, it should be possible to test the actual connection from outside of SAS using a light database client. Testing outside of SAS while configuring is important because it will show you any base configuration problems right away. Troubleshooting these problems from SAS is not easy.

CONFIGURATION IN SAS MANAGEMENT CONSOLE®

Once the OS layer of configuration is successfully completed, you can start the configuration within the SAS Management Console®. Before going forward, it is recommended to test the connection again but this time from a SAS client.

TESTING YOUR INITIAL SERVER DEFINITION

To test the actual connection from SAS, you will need some database credentials. Once available, you can use either a libname statement or proc sql pass-through to test. The libname is a high level test which will ensure that the database can be reached. The pass-through will on the other hand allow you to do a select of a specific table or view. Here is an example of both:

```
/* Libname statement */
Libname TESTLIB <DB2|ORACLE|…> <path|datasrc>='<datasource>'
                               user='<db userid>'
                               password='<db password>'
                               schema='<db schema>' ;

/* Proc sql pass-through */
Proc sql ;
    Connect to <DB2|ORACLE|…>(<path|datasrc>='<datasource>'
                              user='<db userid>'
                              password='<db password>') ;
    create table <test table name> as
    select * from connection to <DB2|ORACLE|…>(
    <test query>
    ) ;
Quit ;
```

A great advantage of using either of these methods at this point is that any problem encountered with the database (such as user rights) will be clearly identified with the database error messages. It is very easy to take these messages and address them directly to the database administrator who will know what to do to fix it.

I order to run these codes, you do not need to have SAS clients installed. If you have SAS Studio enabled, you can use it. It is probably the simplest way of running SAS codes for an administrator. Otherwise, an access to the actual SAS server will grant you access to display manager that you could use for that.

SETTING UP THE SERVER IN SMC

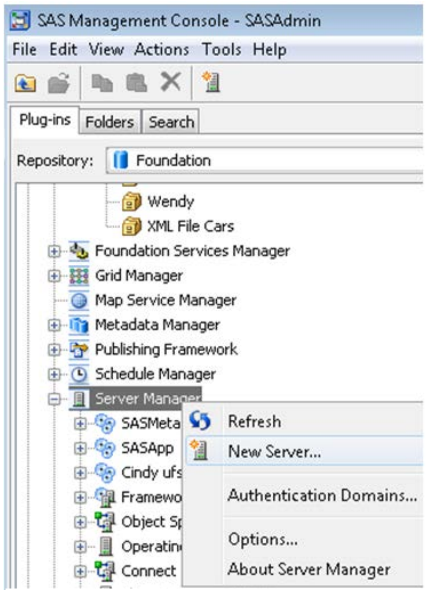


Figure 1. Server Manager plugin from SMC

With our connection tested both outside and inside of SAS, we can continue with the configuration of the second layer.

It consists in creating the SAS server component in the server plugin section. This metadata object is a container for your actual connections. With it, you can specify the database type. You also can document the database version and server name but these two are not used by SAS in any way.

As child objects of this element, you'll find the actual connections. These represent the actual database that you which to map. Every server configuration will make you go through the configuration of a single connection. It is possible to add connections to an existing server at a later time.

While you can create a single server per actual connection, it is recommended to group connections together based on the actual physical server the databases are sitting in.

Consider the example in Figure 2. In this case, it would be possible to create 5 servers each with a single connection. However, the ideal configuration would be 2 servers with each database being a single connection.

When configuring this element, you will have to provide the reference to the database server you have configured in the first layer. The example in Figure 3 is for an ODBC data source but you will have similar information for other databases. You should have one data source per database instance.

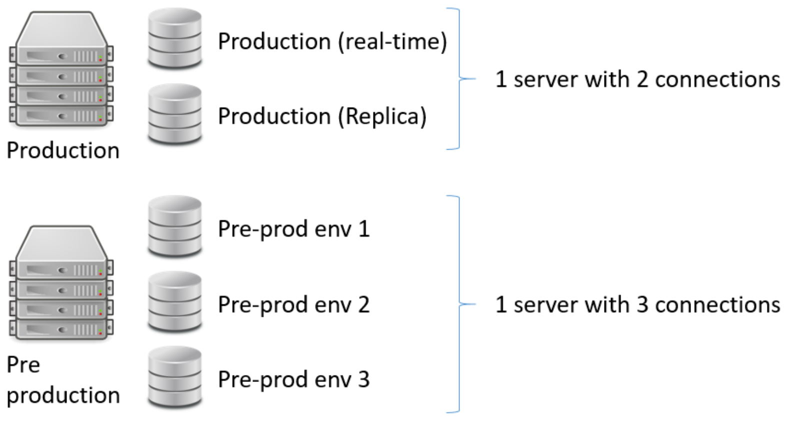
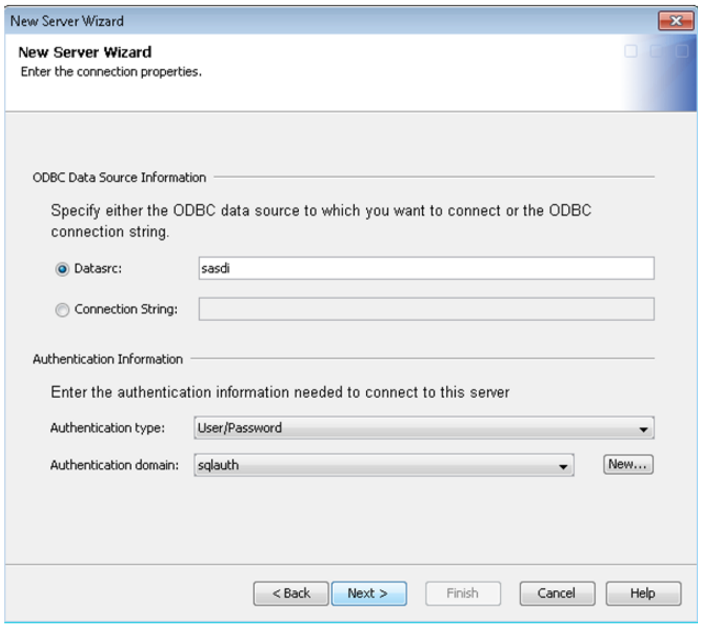


Figure 2. Example of a server environment in SMC

The authentication information allows you to specify how the users will connect to this database. The authentication type should be "User/Password" in most of the cases. This will allow the SAS metadata to store credentials and manage the connections without prompting the user for them. In a very secure environment, it would be possible for a user to not even be aware of his own password but still be able to use it with SAS.

The authentication domain is an important component of the configuration. Overlooking this might lead to complex and hard to maintain SAS environments. This element allows the administrator to label the authentications. This is required as users might have several different authentications so it should be easy to identify which authentication gives access to specific databases. The best approach for this is to create authentications at the server level. Usually, separate instances of databases on a single server will share the same userids. The biggest problem that can arise from using a given authentication on more than one database engine is that when a user wants to change its password on a server, he will need to do the same on every other server sharing that same authentication.


New Server Wizard

New Server Wizard

Enter the connection properties.

ODBC Data Source Information

Specify either the ODBC data source to which you want to connect or the ODBC connection string.

[✓ Datasrc:] sasdi

Connection String:

Authentication Information

Enter the authentication information needed to connect to this server

Authentication type: User/Password

Authentication domain: sqlauth

Figure 3. Configuration of a connection

Once this second layer of configuration is completed, it is possible for SAS to reach the external database. Now a new test can be run to test the final configuration of the server itself. The previous test with SAS code did not interact with the configuration of the server. This one will allow you to ensure that your second layer of configuration was done correctly. To perform this test, it will be important to have an authentication domain assigned to the connection (done through the config) but also have access to a user which has that authentication.

SETTING UP USER AUTHENTICATION

If the authentication type chosen is "user/password", it is a good idea to add this authentication to a SAS user for further testing. To do so, you can simply select a SAS user and then create a new entry in the

Logins tab. Once this entry is added with valid credentials, you can have this user do connection tests with a new version of the SAS code previously shown that will get the credential information from the metadata instead of passing them explicitly.

```
/* Libname statement with authdomain */
Libname TESTLIB <DB2|ORACLE|…> <path|datasrc>='<datasource>'
                               authdomain='<Authentication domain>'
                               schema='<db schema>' ;

/* Proc sql pass-through with authdomain */
Proc sql ;
    Connect to <DB2|ORACLE|…>(<path|datasrc>='<datasource>'
                              authdomain='<Authentication domain>') ;
    create table <test table name> as
    select * from connection to <DB2|ORACLE|…>(
    <test query>
    ) ;
Quit ;
```

This test might not seem be a big leap from the previous one but it is still important as it will ensure that the server and connection configuration are good. It will also confirm that the credentials are accessible by a user with the purpose of connecting an external database. With this validation completed, you could let users connect to the database by using similar codes. This is not very convenient for the users though. This leads us into the configuration of our third and final layer.

CREATING A LIBRARY

The final and more complex layer of configuration is the library. With a library mapped to an external database's schema, the users will be able to use that source without needing to explicitly assign it. This is more convenient for the users but also an actual requirement to some BI tools of the SAS suite that need data to be registered in the SAS management console before being able to use it. The basic library setup is fairly simple; you can launch it from within the Data Library Manager plugin. When creating that new library, you will first be asked which kind of library you wish to create. In this case, you pick your database type of library. It must match the type you picked for the server.

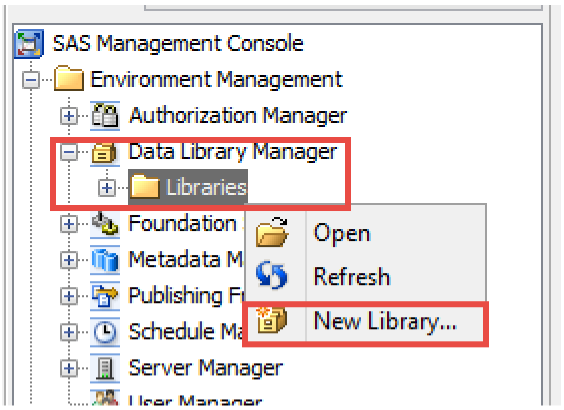
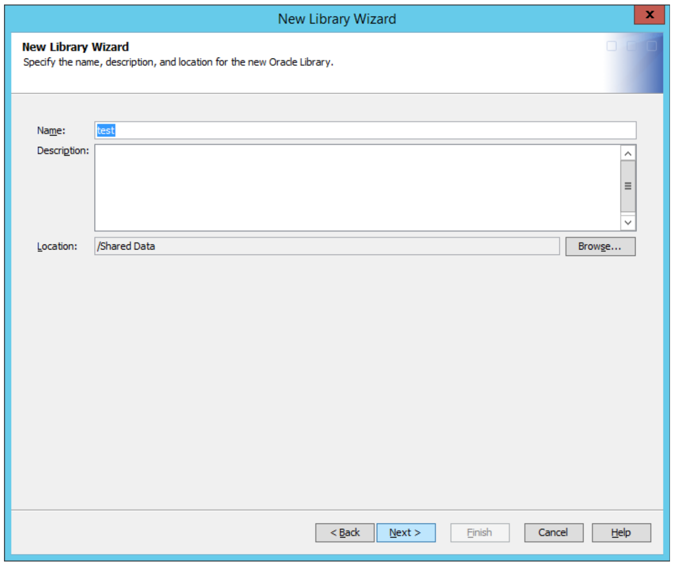


Figure 4. Launching the new library wizard

Name, location and server of the library

The first information to provide for this library is the name. A library actually has two names. One is the legacy name called the libref. This is the name supported with base SAS and older versions. The other is the newer one with less constraints (may contains spaces and more than 8 characters in length). This newer name is simply referred to as "name". Typically, users that write code will use the libref reference and people using tasks from SAS/Enterprise Guide or similar built-in procs will use the newer name. If you have a mixed group of users, it might be easier to set both names with the same value to ease support and communication.

When creating the library, the first input window prompts you for a name, description and location. The location refers to the metadata location. Be careful because the default value for this field is typically the last referenced location regardless of its relevance to libraries. It is recommended to have a folder for each library.



New Library Wizard
Specify the name, description, and location for the new Oracle Library.

Name: test
Description:
Location: /Shared Data

Figure 5. Setting name and location

In the following wizard window, you are prompted to provide the SAS servers for which you want your library to be defined. This is puzzling for most administrators. The main SAS server and often the only choice available is the "SASApp" server (might have a different name based on who installed and configured SAS). Most users (SAS/Enterprise Guide®, SAS/Enterprise Miner® …) will be using libraries through the SASApp server. In most cases, you should specify that server only unless advised otherwise by the software installer or SAS support.

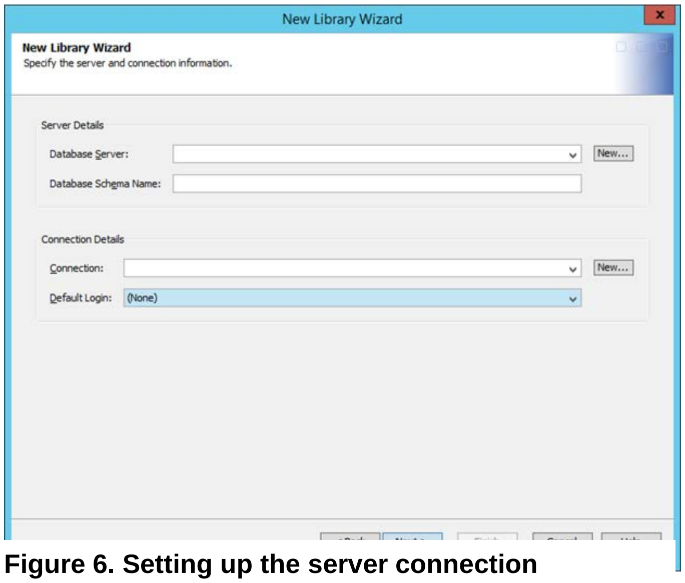
Next is the where you provide the LIBREF for that library. Remember that the libref must respect the base SAS library name restrictions. You have to be creative here since this means 8 characters max with letters, numbers and underscores and it cannot start with a number.

This is also where you could configure advanced options. It is recommended to avoid doing so with the initial config. It is better to troubleshoot problems with a "vanilla" library than it is to do with a customized library. You can always get back to it later and you should actually do it as we'll see later.

Linking to an actual server and schema

The final key portion of the library configuration is the server and connection information window. In here, you will get to link the library to the server defined earlier. The first field is linked to a dropdown menu which allows you to select from the available servers you have defined.

If you did not choose the same library engine for the library as the one you have selected in the server definition, you will not see your server.



New Library Wizard
Specify the server and connection information.

Server Details
Database Server:
Database Schema Name:

Connection Details
Connection:
Default Login: (None)

Figure 6. Setting up the server connection

Once you selected the server, you must provide the name of the database schema. You can only set one schema per library. Then you need to specify the connection you want your library to use. If you have more than one connection for a given server, make sure to select the right one because the initial default value might not be the one you want.

Then you can specify a default login. The default value is "None". In a configuration where we want to define a specific authentication type for the connection, this should be left with the default value.

This concludes the setup of the library using the default advanced options. Often, the users will already have access to it (provided they have the rights and access to an authentication for the connection).

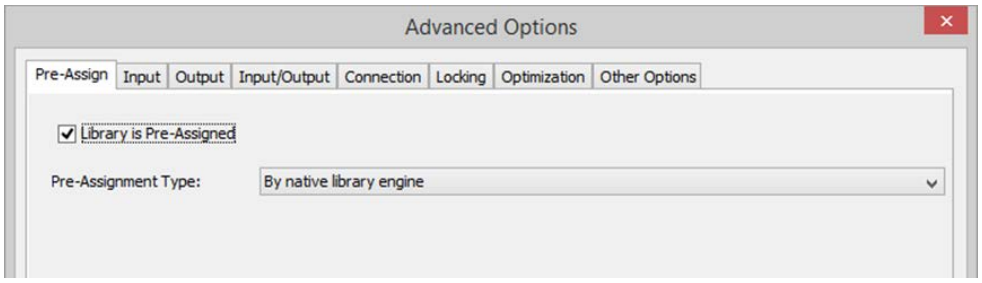
CUSTOMIZING THE ADVANCED OPTIONS

Usually, advanced options are optional and are not critical to the good functionality of a software. In the case of SAS libraries, it is not the same. It is very important to review the advanced options to ensure good user experience and proper security. To access the advanced options, you can simply edit the properties of your library.

The content of these tabs will vary from one database type to another. You may find more options available then the ones listed below.

Pre-Assignment

This single option might be the more important one. Pre-assignment lets you decides whether or not a library is established when the user starts a SAS session or whether he has to establish it manually. Pre-assignment of libraries is really convenient for users and I would advise selecting that option. Now in an environment with lots of external databases connected to SAS, this could maybe impact the startup time. In such case, deactivating the pre-assignment could be considered.



☑ Library is Pre-Assigned

Pre-Assignment Type: By native library engine

Figure 7. Pre-assign options tab

When Pre-Assignment is ticked, you need to select a Pre-Assignment type. The possible values are:

- By native library engines

  This is the default choice. For DBMS libraries, it lets the database handle the security. For connections to external databases, it should almost always be the type chosen.

- By metadata library engine

  This type is used to manage security access at the metadata level. For SAS® base libraries, it is an obvious choice but when it comes to external libraries, you should be cautious. Managing security from the metadata server for an external database should be seen as a sort of filtering rather than actual security. Best practices dictate that the data security should be applied and managed at the source (within the actual database).

- By external configuration

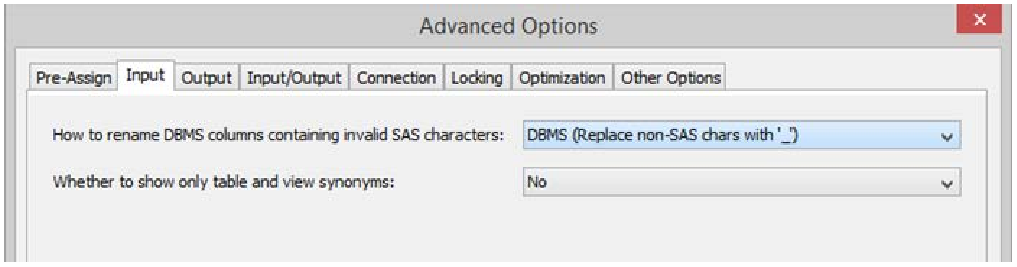
  This type should be considered when you want to move the configuration of libnames outside of the metadata (ex: an autoexec file). A typical usage is within an environment with multiple servers linked with SAS/Connect®.

When it comes to external databases, if you can have the security accesses managed at the database level, you should choose the native library engine for the pre-assignment type.

Input tab

This tab contains the options that will affect how SAS will view the data from the external database. The first option sets the value of the libname option DBGEN_NAME. In most cases, the value 'DBMS' is the best option as it will rename only the invalid characters from column names making these columns easy to recognize.

The second option tweaks the libname option SHOW_SYNONYMS which tells the database to show only synonyms of objects if we want to. These options should be reviewed with your database administrator in order to set them according to the best practices.


Advanced Options

How to rename DBMS columns containing invalid SAS characters: DBMS (Replace non-SAS chars with '_')

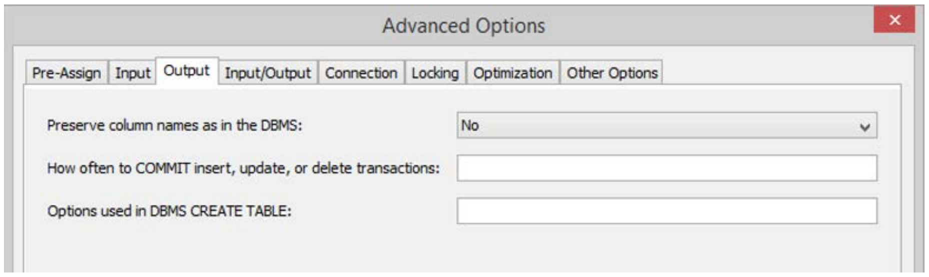
Whether to show only table and view synonyms: No

Figure 8. Input options tab

DB2 offers the possibility of setting the options for CURSOR_TYPE and QUERY_TIMEOUT here as well. The timeout option can be useful with an environment where users can make mistakes and run queries that are too demanding. With it, you can specify a timeout for queries in seconds. Do take into consideration that this setting will apply for everyone so you might have some users for which it is normal to have huge queries.

Output tab

Write access to an external database is not provided nearly as often as read access. In almost all cases, these rights are granted to production batch jobs and power users. For this reason, you should use them rarely. When you need them, make sure to review the options of this tab with your database administrator.


Advanced Options

Preserve column names as in the DBMS: No

How often to COMMIT insert, update, or delete transactions:

Options used in DBMS CREATE TABLE:

Figure 9. Output options tab

The first option lets you set the PRESERVE_COL_NAMES option. This will allow you to create database tables that contains columns with special characters. If this option is set to true, you will have to use it with the system option VALIDVARNAME.

The next two are more self-explanatory. One sets the DBCOMMIT option and the last sets additional database parameters through the DBCREATE_TABLE_OPTS option. Again, do make sure to get the input of your database specialist.

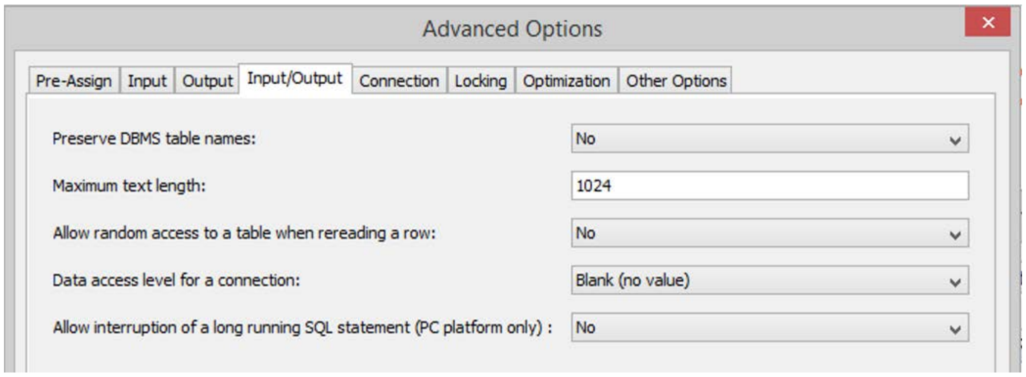
Input/Output tab

Like its title indicates, this section is a mix of input and output options that were not covered in the previous two tabs. In most cases, it indicates that these options affect both cases.

If the database you are connecting has non-standard table names (such as names that contain special characters or spaces), you should enable the first option. It sets the libname option PRESERVE_TAB_NAMES and will allow you to access these tables. Otherwise they won't be available.

The second one is really important as it might truncate data. If you have database tables for which some columns exceed 1024 characters in length, then the default value of the option DBMAX_TEXT will make it truncate at 1024 characters. The maximum value supported by SAS is 32 767.

The third option helps increase performance but should be used with care. It will set the REREAD_EXPOSURE option which allows SAS to work as a random-access engine when re-reading data. This implies that SAS will read specific row numbers when re-reading data. In cases where rows can be deleted, this will generate invalid results.



Preserve DBMS table names: No

Maximum text length: 1024

Allow random access to a table when rereading a row: No

Data access level for a connection: Blank (no value)

Allow interruption of a long running SQL statement (PC platform only) : No

Figure 10. Input/Output options tab

The fourth option will tick the libname option ACCESS for the value read-only if you want to. Even if the credentials used to access a library are read only at the database level, it is better to still enforce it here. This way, if later a userid gets created with write access by mistake, then that person won't be able to delete or replace tables in the database.

The last option of this section only concerns PC platforms. It will allow you to set the value for the access option OR_ENABLE_INTERRUPT. Very little documentation is available on the subject. As with most options, it is recommended to use the default value unless advised otherwise by a SAS support agent or SAS specialist.

DB2 libraries can be set for auto-commit here via a DBMS specific option called AUTOCOMMIT.

Connection tab

This section contains options that will help you control the number of connections to the external database. In a lot of cases, external databases are configured to allow a fix number of concurrent connections. Using the rights settings here will help you void reaching that number.

The first option will allow you to specify whether or not you want SAS to share connections between libraries pointing to a common database. The global option behind is CONNECTION and possible values are the following.

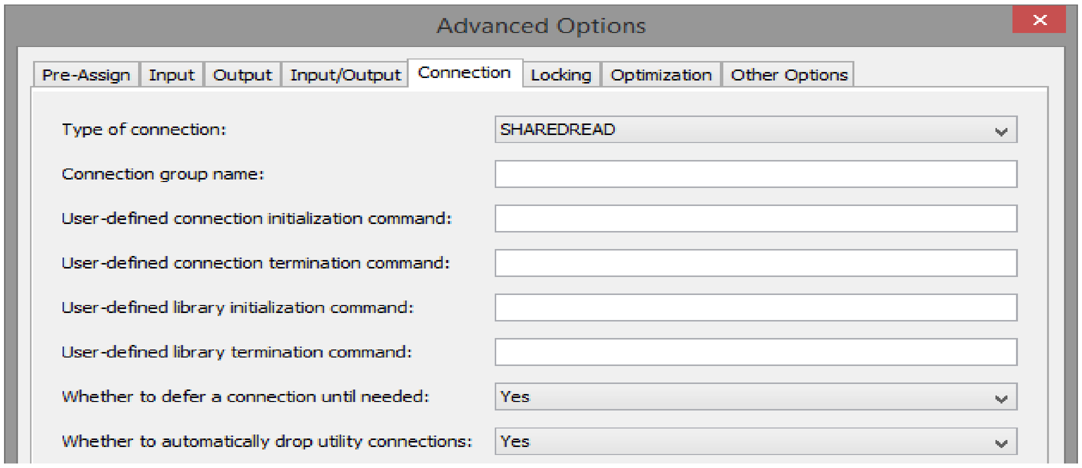
Table 1. Values of the connection option

Value	Description
SHAREDREAD	Whenever possible, READ operations on a SINGLE librefs share a single connection
SHARED	Whenever possible, ALL operations on a SINGLE librefs share a single connection
UNIQUE	Every single connection to the database will generate a single connection
GLOBALREAD	Whenever possible, READ operations across MULTIPLE librefs share a single connection
GLOBAL	Whenever possible, ALL operations across MULTIPLE librefs share a single connection

To limit the number of connections, it is better to use either SHAREREAD or GLOBALREAD. As write access to external databases are few and controlled, this should not in any way generate large number of connections.

The next information is optional. It allows you to explicitly link libraries pointing to a single database together by giving them a group name. This is a kind of label. SAS will perform this in most cases implicitly. Having such libraries linked together as such is crucial for good performance.

SAS can only do pass-through processing on libraries connected to a single database server. The connection group name helps SAS to recognize such library groups.



Advanced Options

Pre-Assign | Input | Output | Input/Output | Connection | Locking | Optimization | Other Options

Type of connection:	SHAREDREAD
Connection group name:	
User-defined connection initialization command:	
User-defined connection termination command:	
User-defined library initialization command:	
User-defined library termination command:	
Whether to defer a connection until needed:	Yes
Whether to automatically drop utility connections:	Yes

Figure 11. Connection options tab

You will then find two pairs of initialization and termination commands. These database specific commands will get executed at specific moments. The first pair will get executed at the start and the end of the connection session. The second is executed at the start and end of the library session. These options would most likely be requested by your database administrator. They will help you configure how SAS handles the connection.

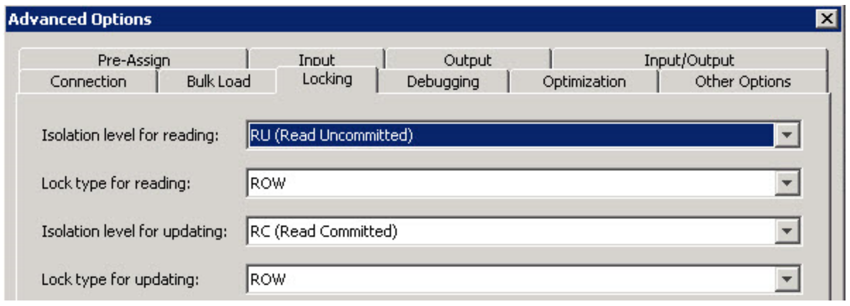
The defer option is extremely valuable. It will instruct SAS to hold the library assignment until fist used. This might slightly increase the overall SAS initialization time for users. However, the greatest gain is that connections being established are going to be used. Without it, every single DBMS library will need an active connection to the database whether it is going to be used or not during the session.

The last option is available for some external databases that work with utility connections (such as DB2 and Netezza). In these cases, utility connections can be spawned while running queries. This option allows you to set a value to the UTILCONN_TRANSIENT global option. With it, you can inform SAS whether to keep the utility connections active for future use or to delete them as soon as the work is completed. Unless performance issues are encountered with such connections being establish, it is recommended to set the value to ‘YES’ so that you don’t keep unused connections active.

Locking tab

The locking of rows during operations to the database is a very important subject in the data warehousing operations. It allows to lock certain rows of data (or even the whole table) so that they can’t be altered or even viewed while they are being read of updated.

In most cases, you will connect SAS to read-only information database. For these libraries, you should make sure that the action of reading data won’t lock any production job from updating it. To do so, you should adjust theses settings here. There are Isolation levels for both reading and updating. The recommended option is to specify “RU (Read Uncommitted)” for reading. For updating, “RC (Read Committed)” makes more sense but you have to specify when committing your changes.



Isolation level for reading:	RU (Read Uncommitted)
Lock type for reading:	ROW
Isolation level for updating:	RC (Read Committed)
Lock type for updating:	ROW

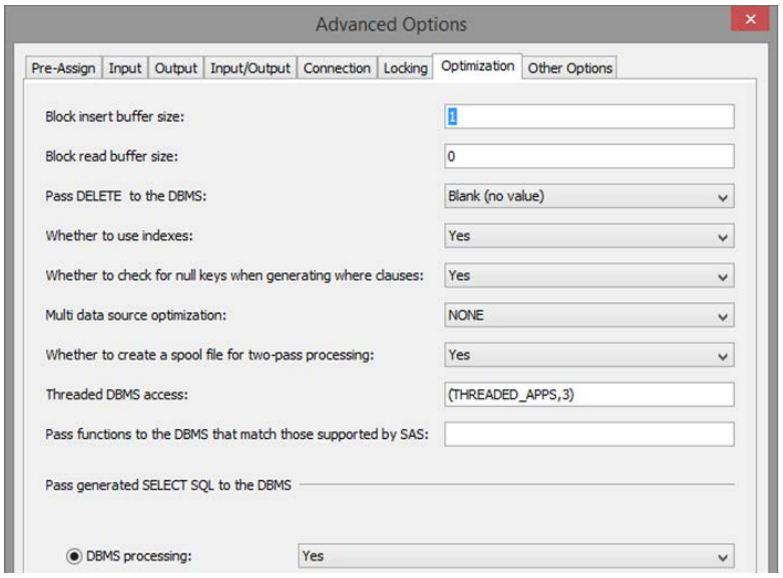
Figure 12. Locking options tab

Optimization tab

This panel contains information that will allow you to adjust very specific performance options. You should use care while changing these values as they can work with some usage of the library and not with others. You should get advices from your database administrator for the options to tweak.

The block insert buffer and read buffer allows you to specify the number of rows of data to be handled at one time. A common misconception is that you need to set a high value in there all the time. Performance will vary from one database to another and from one table to another (ex: very wide table vs thin table).

The pass delete is important to set if you plan on deleting data from tables. If activated, it will produce the libname SAS option “DIRECT_EXE=DELETE” to be used. This will instruct to pass over the database any delete instruction being run. Without this option, SAS would have to read the entire table and then delete one row at a time.



Block insert buffer size:	1
Block read buffer size:	0
Pass DELETE to the DBMS:	Blank (no value)
Whether to use indexes:	Yes
Whether to check for null keys when generating where clauses:	Yes
Multi data source optimization:	NONE
Whether to create a spool file for two-pass processing:	Yes
Threaded DBMS access:	(THREADED_APPS,3)
Pass functions to the DBMS that match those supported by SAS:	

Pass generated SELECT SQL to the DBMS

● DBMS processing:	Yes

Figure 13. Optimization options tab

The next option helps you improve performance while joining large tables from an external database with a small SAS table. It will make a better use of database indexes.

Running queries against mixed sources (such as a DBMS library and SAS table) can be hazardous. In most cases, it will force SAS to handle the query thus making huge extracts from the database into temporary tables. These operations should be limited and optimized manually.

The option that allows you to check for null keys when generating WHERE clauses will set the libname option DBNULLKEYS. The default value will vary from one database to another. It is recommended to keep the default value here.

The Multi data source optimization option allows you to instruct SAS to create “in” clauses to optimize queries. This process will take a small table joined by the query and replace that join by generating an “in” clause instead. It is recommended to use the default value for your DBMS. These “in” clauses are not suited for every query and therefore general options like this one should be handled with care.

The option on spool file will instruct SAS to create a utility spool file when reading data more than once. Some queries will ask of SAS to read over again data. In such cases, this option will ensure that no additional database access is needed and that the data is exactly the same. The only downside to this option is the potential size of the spool file created. Proc SQL does not limit the space used by spool files and these can get large at time with complex queries. Do keep in mind that the size of your workspace might have to be increased.

The Threaded DBMS access lets you specify the scope of threaded reads and the number of threads. This option will directly affect the performance of the external database. You should keep the default values but if you need to change them, make sure that these settings are reviewed by your database administrator.

The “Pass functions to the DBMS” will allow SAS to pass over to the DBMS SAS functions that matches the database native functions. By default, no functions are specified and it is most likely the best option too. According to SAS, this option can cause unexpected results, mostly because of the processing of NULLs by the databases. Use only with great care and limited scope if you must.

The final option lets you specify which kind of processing is passed over to the database. In most cases, you should leave the default value of “YES” as it will allow SAS to generate DBMS specific queries and send it over. Changing this value will cause SAS to hold up on some of these queries and handle a portion by itself. Doing so will almost always imply extracting more data than required to handle the query.

Other Options tab

The options just covered in these tabs are a portion of the DBMS parameters that you can set. Every database comes with a set of options that can be customized. When going through the SAS/Access documentation for your DBMS, if you find an option you wish to use that isn’t in the previous tabs, you can set it in this final tab.

Using this functionality should be rare. However, if a database administrator or SAS specialist advises you to set a specific option not listed then it should be considered.

OTHER CONSIDERATIONS

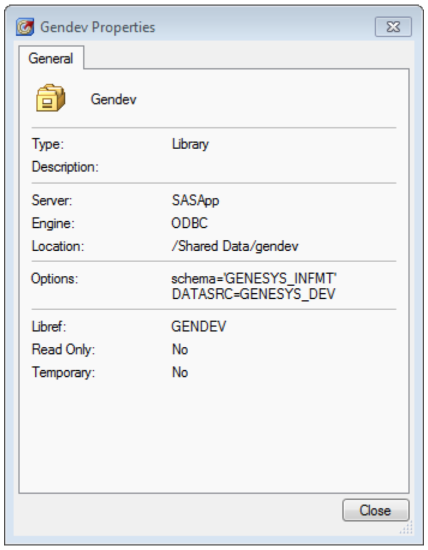
The previous sections have covered in depth how to create libraries to external databases. There are still a few important topics not covered. Here are a few points to take into consideration while building a SAS® ecosystem connected to external databases.

USING CONSTANCY IN THE CONFIGURATION

Having a few schemas connected to libraries in SAS with a common connection group name is not enough to ensure pass-through processing. All of the libraries to a single server should have the same set of advanced options. For example, having one library marked for “Read Uncommitted” and another with “Read Committed” will force SAS to handle the queries on these two libraries within the SAS environment.

It is possible to test queries with the use of the SAS option “OPTIONS SASTRACE=',,,d' SASTRACELOC=SASLOG NOSTSUFFIX;” to get details on how a SQL code is handled by SAS. This will allow you to see what query is actually generated for DBMS processing.

To compare two libraries easily, you can also either display the libname instruction from the SMC. Alternatively, in SAS/Enterprise Guide® you can select a library and select “properties” from the right mouse button menu to get the connection information as shown in Figure 14.



Gendev

Type:	Library
Description:	
Server:	SASApp
Engine:	ODBC
Location:	/Shared Data/gendev
Options:	schema='GENESYS_INFMT' DATASRC=GENESYS_DEV
Libref:	GENDEV
Read Only:	No
Temporary:	No

Figure 14. Library properties

EDUCATE USERS

From an administrator’s point of view, the libraries are easy to identify. However, from a user’s perspective, they all appear the same way in the client tools (SAS/Studio®, SAS/Enterprise Guide® …). This can be a problem as for them, when they are joining data from different libraries, they are joining SAS® tables.

For that reason, education is important. Users should know which libraries contains SAS tables and which points to DBMS tables. Also, in an environment with more than one DBMS type, they should know which is which (ex: DB2 vs Oracle). This might limit the number of people running huge queries across multiple sources.

A naming convention for libnames and servers is a great and easy way to start educating users. If the libraries are created with names which indicates where they are located, it will help users understand better the nature of the underlying data. Users having good knowledge of the libraries might not prevent them from running bad queries but it will at least filter out the ones that do it unknowingly.

Teaching advanced users to use pass-through queries for very specific tasks is also a good approach. Some cases might be better handled by sending an explicit query to the DBMS instead of having SAS generate one for you. With these, you will also get the possibility of creating temp database tables with the use of “with” database statement (for DBMS that support it).

CONCLUSION

Although it looks complex, the base configuration of a library to an external database from the bottom up is fairly simple with the configuration wizards available. The complexity really arises from the wide range of advanced options available and also from the difficulty associated with troubleshooting connection issues from SAS.

By building your configuration to external databases by layers, you can ensure to identify and address problems early. This approach should always be used to ensure constancy and also simplicity for new administrators.

Adjusting advanced options really needs to be done with the proper care. Setting up a library might only take a few minutes but problems generated by wrong options used will monopolize hours of your time along the way in the form of troubleshooting.

Finally, when building metadata content such as servers and libraries, do take into consideration the long time support of that structure. Try to maintain standards which will be easier to maintain and understand.

Documentation for metadata structure is rarely done by administrators due to time constraints so having a simple environment which uses standards will ease the learning curve of new people.

REFERENCES

Nelson, Gregory. “Pre-assign SAS libraries? If so, which method?” May 15, 2013. Available at http://blogs.sas.com/content/sgf/2013/05/15/pre-assign-sas-libraries-if-so-which-method/.

SAS® Institutes. “Accessing DB2 Data with SAS9.” Available at DB2 connection information: http://www.sas.com/partners/directory/ibm/db2_sas9.pdf.

SAS® Institutes. “SAS(R) 9.4 Intelligence Platform: Data Administration Guide, Fifth Edition.” Available at http://support.sas.com/documentation/cdl/en/bidsag/68193/HTML/default/viewer.htm#p1icta556r8bacn1ns9kvvnxd4iv.htm.

SAS® Institutes. “SAS/ACCESS® 9.4 for Relational Databases: Reference, Ninth Edition.” Available at http://documentation.sas.com/?cdcId=pgmsascdc&cdcVersion=9.4_3.3&docsetId=acreldb&docsetTarget=titlepage.htm.

SAS® Institutes. “SAS/ACCESS(R) 9.4 Interface to PC Files: Reference, Third Edition.” Available at http://support.sas.com/documentation/cdl/en/acpcref/67382/HTML/default/viewer.htm#n0yx5o6hz6nts6n0zi844i9fsc2l.htm.

CONTACT INFORMATION

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

Mathieu Gaouette
Prospective MG inc.
mathieu.gaouette@prospectivemg.com
http://www.prospectivemg.com