CHAPTER

9

Using Remote Library Services (RLS)

Introduction

SAS/CONNECT (single-user) or SAS/SHARE (multi-user) software gives you access to data by means of *remote library services* (RLS). This access, which includes updating, is given to data that is stored in SAS data files, PROC SQL views, SAS/ACCESS views, DATA step views, MDDBs, FDBs, DMDBs, external databases, or SAS catalogs. However, updates to data in an external database, such as ORACLE, can only be made through a single-user server.

RLS requires use of a program-to-program communications access method, such as:

□ APPC

□ DECnet

□ NetBIOS

□ TCP/IP.

If the local and remote hosts have the *same* internal data representations, RLS allows you to access SAS data sets, PROC SQL views, SAS catalogs, and external databases that are defined by SAS/ACCESS librefs. SAS/ACCESS librefs are created by using Version 7 or Version 8 of the SAS/ACCESS software.

However, if you intend to transfer SAS files between two Version 7 or later SAS sessions, you may be able to take advantage of the cross-environment data access (CEDA) facility. This facility is an alternative way to access data that has been written with a different host representation. To find out the scenarios when you use CEDA, see Chapter 13, "Using Cross-Environment Data Access (CEDA)," on page 93

Note: Before using single-user RLS, you must establish a connection between a local and a remote host by using the SIGNON statement. See Chapter 23, "Starting and Stopping SAS/CONNECT Software," on page 193 for more details. △

Benefits of RLS

If you need to maintain a single copy of the data on the remote machine and keep the processing on the local system, then RLS is the correct choice. In general, RLS is the best solution if

- □ the amount of data that is needed by the local processing is small.
- □ the remote data is frequently updated.
- □ your data center rules prohibit multiple copies of data.

RLS allows you to access your remote data as if it were local. This feature eliminates an explicit step of coding an upload or download of the data before processing it. It also permits the GUI of an application to reside on the local system while the data remains in the remote environment (for example, a local FSEDIT session of a remote data set). Applications can be built that provide seemingly identical access to local and remote data, without requiring the end user to know where the data resides.

Using RLS, it is possible to perform updates to data in external databases. RLS enables a single user who is accessing data that is stored in an external database to perform updates to that data through the use of the single-user server. This allows you to update remote data in an external database as a result of local processing.

Considerations for Using RLS

To make the best use of RLS, careful thought must be given to the following:

- □ the amount of data that will be accessed by your application
- □ whether you need multi-user or single-user data access
- □ whether your application will be making multiple passes of the data
- □ the resulting effects on your network load.

Analyzing these criteria will help you determine when to use RLS and when to use data transfer or compute services. This section describes situations in which RLS may not be the ideal way to access your data and suggests alternative solutions.

Accessing data through remote library services is not efficient for large amounts of data. In this case it is usually better to use compute services to move the processing to the remote system or to use a combination of compute and data transfer services.

Similarly, RLS is not efficient for multiple passes of the data. Although the data moves from the remote machine to the local machine, the data is not written to the local disk. If the procedures that are run make multiple passes of the data, or the entire procedure must be run more than once against the data, it is better to transfer a copy of the data to the local machine. You incur the network traffic cost only one time rather than paying the cost for each pass of the data.

Data transfer services may also be a better choice when response time is delayed. This situation can occur if you are accessing remote data that is being updated simultaneously by other users. If delayed response time cannot be tolerated, consider transferring a copy of the data to the local system and removing the live data from your application.

It is very important to remember that the data you access by means of RLS must still move through the network in order to be processed by the local execution. As a result, you need to program your application to minimize the amount of data that is being requested by the local processing.

The main difference between data transfer services and RLS is that by using data transfer services, after the data has moved through the network to the local system, it is written to disk and is available for subsequent local processing.

In contrast, RLS moves data through the network to the local system where it is processed by the local execution and then discarded. Subsequent analyses of the same data would require the data to be moved through the network each time the local processing requested the data. Therefore, a client/server application should be designed to balance the costs of

- moving and storing data with data transfer services.
- moving data with remote library services.
- computing on the remote host.
- computing on the local host.

Types of Data Accessible through RLS

Data that is stored in any of the following is accessible by means of RLS across all hosts:

- SAS data files
- PROC SQL views
- SAS/ACCESS views
- data defined by a DATA step view
- MDDBs
- FDBs
- DMDBs
- SAS catalog*.

If you attempt to access a SAS catalog when your local and remote SAS sessions run on machines that have different internal representations of data, you receive the following error message:

```
ERROR: You cannot open <catalog name> through
       server <rsessid> because access to
       catalogs is not supported when the user
       machine and server machine have different
       data representations.
```

Using a single-user server, RLS provides update access to data that is stored in external databases. The SAS/ACCESS engines, as well as the SQL engine, recognize the single-user server as one user and, therefore, enable update mode for external database sources. However, these same engines prohibit update access to external database sources when using RLS with a multi-user server. Updating is prohibited because it would be impossible for either a multi-user server or a database product to detect conflicting requests from multiple users and, therefore, impossible to guarantee data integrity and security or to provide audit trails.

You can also use RLS with SAS data views. SAS data views include SAS/ACCESS views, DATA step views, and PROC SQL views. A *view* is a SAS data set that describes other data. It is processed as a data set by an engine that reads the underlying data and uses the description to return the data in the requested form. This process is called *view interpretation*.

When the library that contains the view is accessed through a server, the view is interpreted in the server session by default. This means that the engine is loaded and called by the server to read and transform the underlying data, and a minimum amount

* Catalog access is not supported if the local and remote platforms have different internal representations of data.

of data is moved through the network. The local processing is unaware that a view is involved.

If the view is a PROC SQL view or if the local and remote host architectures are the same, you can cause the view to be interpreted in the local session. This is done by specifying RMTVIEW=NO in the LIBNAME statement that is used to define the remote library. If the architectures are not the same, SAS/ACCESS views and DATA step views can only be interpreted in the remote session.

SAS files of type PROGRAM cannot be accessed cross-machine. These files contain compiled DATA step code, which you cannot execute locally. The DATA step can be executed in the server's execution if it is referenced by a DATA step view that is interpreted there.

If you attempt to access a SAS utility file of either type ACCESS or PROGRAM when your local and remote SAS sessions are running on machines that have different internal representations of data, you receive the following error message:

```
ERROR: You cannot open <utility file name> through
       server <rsessid>, because access to utility
       files is not supported when the user machine
       and server machine have different data
       representations.
```

RLS Restrictions for Short Numerics and Mixed-Type Variables

SAS data sets that can be accessed by means of RLS across different host internal data representations should not include two-byte numeric variables. Due to their underlying floating-point representation scheme, most hosts on which the SAS System runs have a minimum numeric variable length of 3.

A data set that contains a two-byte variable cannot be accessed by using RLS from a three-byte host. However, this type of data set could be accessed by using the data transfer services of SAS/CONNECT. The UPLOAD and DOWNLOAD procedures can be executed so that they promote the length by one byte of any numeric variable that has a length that is less than 8 before transferring the variable to the target host. In order to promote the length of the short numerics, use the EXTENDSN=YES and V6TRANSPORT options in either the PROC UPLOAD or the PROC DOWNLOAD statement. See Chapter 17, "The UPLOAD Procedure," on page 107 and Chapter 18, "The DOWNLOAD Procedure," on page 129 for more information about these options.

With clever programming in the DATA step, it is possible to store numeric values in character variables and store character values in numeric variables. These programming techniques must not be used in any data set that is accessed by means of RLS. RLS performs character translation on character variables and numeric translation on numeric variables. As a result, the value that the programmer expected is not preserved. There are no means to detect such usage. Therefore, no message is issued.

The Implications of Data Translation for RLS

When accessing data with RLS, translation occurs on numeric variables only if the local machine and the remote machine represent floating-point numbers differently. Translation occurs on character variables only if their character representations differ. Values are translated directly from the source representation to the target representation; they do not pass through transport format.

Especially with numeric variables, translation from one representation to another may alter the value of the variable. The common type of alteration is loss of precision. This occurs when the source representation uses more bits to represent the mantissa than the target representation. There is no warning produced about loss of precision during translation.

The data transfer services of both the UPLOAD and DOWNLOAD procedures preserve precision by promoting the length by one byte of any numeric variable with a length that is less than 8 before transferring it to the target host. However, this is NOT the default action in Version 7 or Version 8. The EXTENDSN=YES and the V6TRANSPORT option must be used in order for the length to be promoted.

A rare type of value distortion is loss of magnitude. This occurs when the source representation has a greater exponent range than the target representation, and a value with a magnitude that lies in the excess range of the source representation is translated. Of course, the magnitude of this type of distortion is potentially very great, as is the percentage change. The following warning is produced when this type of alteration occurs:

```
WARNING:  The magnitude of at least one
          numeric value was decreased to
          the maximum the target
          representation allows, due to
          representation conversion.
```

Refer to the description of numeric precision in *SAS Language Reference: Concepts* for a detailed discussion about the numeric representation of SAS variables.

Using Data Views with Servers

The Remote Library Services (RLS) features of the SAS System can be used with three types of views:

□ SAS/ACCESS views

□ DATA step views

□ PROC SQL views.

A view is a SAS file that contains no data, but it describes other data. A view is processed by an engine that reads the underlying data and uses the description to return the data in the requested form. This process is called *view interpretation*.

When the library that contains the view is accessed through a server, the view is interpreted in the server's session by default. This means that the engine is loaded and called by the server to read and transform the underlying data. Only a small amount of data is moved through the network, and the local processing is unaware that a view is involved.

If the view is a PROC SQL view or if the local and remote machine architectures are the same, you can cause the view to be interpreted in the local session. This is done by specifying RMTVIEW=NO in the LIBNAME statement that is used to define the remote library. If the architectures are not the same, SAS/ACCESS views and DATA step views can only be interpreted in the remote session.

Interpreting a view as data can produce significant processing demands. When a view is interpreted locally, that frequently means that a lot of data has to flow to the local session. This removes processing demands from the server session but increases network load, which is usually not desired.

Recommendations for PROC SQL Views

PROC SQL views are especially good candidates for interpretation in a remote SAS session if

- □ the number of observations that are produced by the view is much smaller than the number of observations that are read by the view.
- □ the data sets that are read by the view are available to the server.
- □ the amount of processing necessary to build each observation is not large.

Conversely, PROC SQL views should be interpreted in users' SAS sessions if

- □ the number of observations that are produced by the view is not appreciably smaller than the number of observations that are read by the view.
- □ some of the data sets that are read by the view can be directly accessed by the users' SAS sessions.
- □ the amount of processing that is done by the view is considerable.

Using WHERE Processing to Reduce Network Traffic

When using RLS, one of the best ways to reduce the amount of data that needs to move through the network to the local process is to use WHERE statement processing whenever possible. When WHERE statements are used, the WHERE clause is passed to the remote environment, interpreted, and only the data that meets the selection criteria is transferred to the local environment for processing.

If the data you are accessing is stored in an external database, the WHERE statement is passed to the database and evaluated, if possible. If the database cannot complete the evaluation, the server completes it before returning any of the data to the REMOTE engine in the local session. Examples 2, 4, and 6 in Chapter 11 show how to use WHERE statements.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/CONNECT User's Guide, Version 8*, Cary, NC: SAS Institute Inc., 1999. pp. 537.

SAS/CONNECT User's Guide, Version 8

ISBN 1–58025–477–2

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

1st printing, September 1999

The Institute is a private company devoted to the support and further development of its software and related services.