CHAPTER

7

Using SAS/SHARE to Access Remote Files

Supported Members and External Files 55
Accessing Remote SAS Files by Using a SAS/SHARE Server 56
Requirements of a Typical Client/Server SAS Session 56
  Server Host 56
  Client Host 57
Role of the SAS/SHARE Server in a Client/Server Session 57
Deciding Where Views Should Be Interpreted 58
Managing Incompatible Client/Server Host Types 58
  Numeric Translation 59
  Character Translation 59
Accessing Version 6 Files or Version 8 Files in a Mixed Library 60
Concatenating Multiple Libraries 61
Accessing Files in a Cross-Version Environment 62
  Accessing Data Sets in a Cross-Version Environment 62
  Accessing DATA Step and PROC SQL Views in a Cross-Version Environment 63
  Accessing SAS/ACCESS Views in a Cross-Version Environment 63
  Accessing Catalogs in a Cross-Version Environment 63

Supported Members and External Files

You can use SAS/SHARE to access any of these members or external files:

- □ Data set
- □ Catalog and entry
- □ DATA step view
- □ PROC SQL view
- □ SAS/ACCESS view
- □ MDDB (Multi-dimensional Database) file.

SAS/SHARE also supports access to certain third-party database files by means of its Remote SQL Pass-Through facility. You must have SAS/ACCESS software licensed on the remote computer for that database.

For complete details about relational databases, see *SAS/ACCESS Software for Relational Databases: Reference*. For details about nonrelational databases, see *SAS/ACCESS Interface to CA-DATACOM/DB: Reference*, *SAS/ACCESS Interface to IMS-DL/I Software*, *SAS/ACCESS Interface to CA-IDMS Software: Reference*, or *SAS/ACCESS Interface to SYSTEM 2000 Data Management Software: Reference*, as appropriate.

Accessing Remote SAS Files by Using a SAS/SHARE Server

An alternative to accessing remote SAS files (including Version 8 CEDA files) through the network, the SAS/SHARE server provides concurrent multi-user read, write, and update access to common data that is located on a remote host. The SAS/SHARE locking facility enables the SAS/SHARE server to manage complex requests for I/O service. For complete details about SAS/SHARE, see *SAS/SHARE User's Guide*.

A fundamental feature of SAS/SHARE is its ability to manage requests for access to common data among multiple hosts that may or may not be compatible with the other. For incompatible hosts that request access to common data, SAS/SHARE performs a dynamic translation of the incompatible data between the native formats of the hosts for each read, write, or update operation.

Unlike the traditional transport methods that create an explicit transport file, SAS/SHARE dynamically translates only the incompatible data per transaction. This dynamic translation bypasses the explicit creation of a transport file. For example, if a Version 6 Macintosh client reads data on a server that is on an HP-UX host, only the character data needs translation. The Macintosh and the HP-UX hosts represent numeric data in the same way; however, they represent character data differently. For lists of compatible hosts, see Chapter 16, "Architectural Compatibility," on page 119.

Note: For the MDDB member type, the SAS/MDDB server rather than the SAS/SHARE server dynamically translates the data upon access. △

Requirements of a Typical Client/Server SAS Session

In order to establish a pathway between a local SAS/SHARE client and a remote SAS/SHARE server, operations must be performed at both the client and the server hosts.

Note: We use the terms *client* and *server* to refer strictly to SAS/SHARE clients and SAS/SHARE servers. Other types of clients such as an htmSQL client, a Java applet, or application client are not discussed here. △

Server Host

Here are the basic tasks that a server administrator performs to start a server:

1 Specify a communications access method that a client uses to access the server.

Note: Do not confuse the communications access method that is used with SAS/SHARE and SAS/CONNECT with the access method that is used as an argument to the SAS FILENAME statement. △

2 Start a server.

Tasks that you must perform vary by host and access method. Examples of such tasks are setting SAS system options to maximize server performance and to configure security features. For complete details, see *Communications Access Methods for SAS/CONNECT and SAS/SHARE Software.*

Here is a simple example of how to start a server on a UNIX host:

```
options comamid=tcp;
libname records '/hist101/term1';
proc server id=shr1 authenticate=optional;
```

```
run;
```

This example declares the TCP/IP communications access method as the pathway between server SHR1 and an accessing client. A pre-defined server library is made available to clients for subsequent access. Server SHR1 must also be declared in the UNIX **/etc/services** file.

Client Host

Here are the basic tasks that an end-user or an applications programmer performs to access data through a server:

1 Specify a communications access method that a client uses to access the server.

2 Specify a library for access by means of the server.

Like the server session, the client session also requires the user to perform tasks that vary by host and access method. Examples of such tasks are setting a SAS system option to maximize server performance and to configure security features.

Here is a simple example of how USER1 might access data by means of SHR1:

```
options comamid=tcp;
libname records server=rmtnode.shr1;
```

This example declares the TCP/IP communications access method as the pathway between the server SHR1 and an accessing client. The pre-defined server library RECORDS is accessed by means of the server SHR1 that is located on UNIX host RMTNODE.

Role of the SAS/SHARE Server in a Client/Server Session

Here is a simple example of how a SAS/SHARE server manages multi-user access to a common data set.

1 Following the setup of a pathway to the server SHR1, client USER1 on a Windows host creates a data set named TERM1 in the server-defined library RECORDS:

```
data records.term1;
  input student $ test1 test2 final;
  datalines;
Barney 80 74 88
Betty 98 89 92
;
proc print data=records.term1;
 run;
```

2 USER2 on a CMS host starts another SAS session in order to edit the data set TERM1 in the server-defined library RECORDS. Here is typical SAS code:

```
options comamid=tcp;
libname records server=rmtnode.shr1;
proc fsedit data=records.term1;
run;
```

This example declares the TCP/IP communications access method as the pathway between the server SHR1 and the client USER2. The pre-defined server library RECORDS is accessed by means of the server SHR1 that is located on the UNIX host RMTNODE. The FSEDIT procedure accesses the first observation in the data set that USER1 created. The server SHR1 locks the first observation in the data set to USER2. USER2 then changes a grade in the first observation.

3 USER1 also attempts to edit the first observation in the data set TERM1 by using these statements:

```
proc fsedit data=records.term1;
run;
```

Because the server locked the first observation in the data set for USER2's FSEDIT session, USER1 cannot gain access to that observation for update. Only after USER2 saves the change and moves to the next observation or exits the FSEDIT session, does the server release the lock, which permits USER1 to update the first observation.

Deciding Where Views Should Be Interpreted

You decide whether a view (either PROC SQL, DATA step, or SAS/ACCESS) is interpreted in the client session or the server session. Factors to consider are:

Location of data

The view must be interpreted at the location that has access to the data.

Performance

If the data is equally accessible to the client and the server, consider the amount of data that must be sifted in order to produce the view. If a view selects most of the data, interpreting on the client may be preferable. If a view selects a small amount of data, interpreting on the server may be preferable.

Version of SAS that runs in the client session and the server session

If the server and client run different versions of SAS, and both have equal access to the SAS data, consider interpreting the view at the location that requires no data conversion. For example, if a Version 6 client accesses a Version 8 SAS data set by means of a Version 8 server, interpreting the view at the server is preferable because the data set does not have to be converted at the server. However, the view would have to be converted before it could be interpreted at the client.

You use the RMTVIEW= option in the LIBNAME statement to specify the location for the view interpretation. RMTVIEW= takes the values YES (interpretation is done at server) or NO (interpretation is done at client). You specify the RMTVIEW= option in the LIBNAME statement in either the client session or the server session. Here is an example:

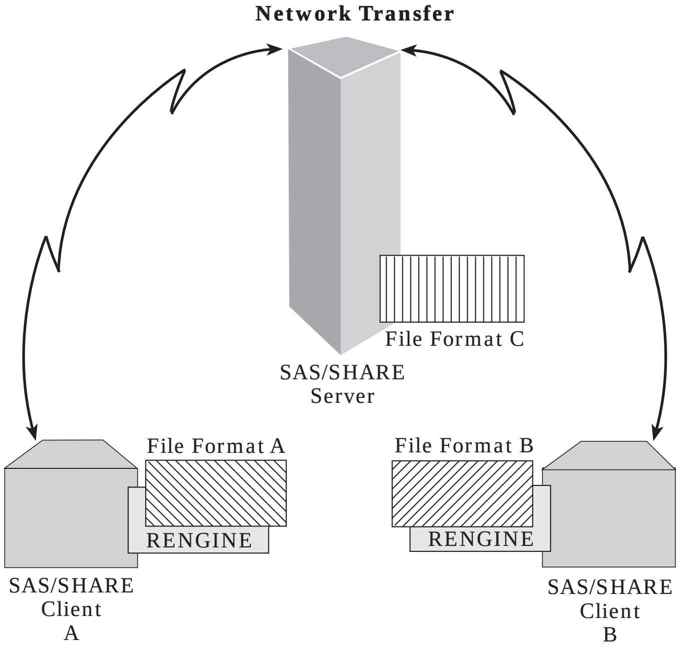
```
libname grades 'SAS-data-library' server=shr1 rmtview=yes;
```

In this example, the view is interpreted in the server session.

Managing Incompatible Client/Server Host Types

SAS/SHARE dynamically translates the client/server host native formats for each client/server transaction, bypassing the explicit creation of a transport file. However, the server does not perform this task. Instead, each client's REMOTE engine is capable of performing its own data translation when interacting with the server. A client automatically invokes a remote engine by specifying the SERVER= option in the LIBNAME statement. Figure 7.1 on page 59 shows that the client REMOTE engine reconciles each client/server transaction.

Figure 7.1 Reconciling Incompatible Client/Server Host Formats by Using the Client REMOTE Engine



Numeric Translation

If the numeric representation of the SAS/SHARE client host and the server host are incompatible, the client REMOTE engine performs a translation per client/server transaction. Incompatible numeric representations cause these numeric data problems:

loss of *precision*
: occurs when the source host uses more bits to represent the mantissa than does the target host.

loss of *magnitude*
: occurs when the source host has a greater exponent range than does the target host.

No error or warning message for these conditions is displayed during numeric translation. For a complete description of the numeric representations of SAS variable values, see *SAS Language Reference: Dictionary*.

Character Translation

EachVersion 8 host supports one of the following character representations:

□ EBCDIC

□ ACSCII-OEM

□ ASCII-ANSI

□ ASCII-ISO.

Note: Only Version 6 supports the ASCII-MAC character representation on the Macintosh host. △

For a list of Version 8 hosts that support each of these character representations, see "Character Architectural Groups" on page 121.

If the character representations on the SAS/SHARE client host and the server host are incompatible, for each client/server transaction the client REMOTE engine translates incompatible characters by first consulting the SASUSER.PROFILE and then the SASHELP.HOST catalogs for the appropriate catalog entry of type TRANTAB. Table 7.1 on page 60 lists the default character translation tables and the catalog entries that store them.

Table 7.1 Default Character Translation Tables

Translation Table	Catalog Entry Name
EBCDIC/ASCII-ISO	_0000030
EBCDIC/ASCII-ANSI	_0000060
EBCDIC/ASCII-OEM	_00000A0
EBCDIC/ASCII-MAC	_0000120
ASCII-ISO/ASCII-ANSI	_0000050
ASCII-ISO/ASCII-OEM	_0000090
ASCII-ISO/ASCII-MAC	_0000110
ASCII-ANSI/ASCII-OEM	_00000C0
ASCII-ANSI/ASCII-MAC	_0000140
ASCII-OEM/ASCII-MAC	_0000180

Each catalog entry contains a translation table for importing and a table for exporting. For example, the EBCDIC/ASCII-OEM table on an OS/390 SAS/SHARE client means:

Import
: To read data from the server, the client remote engine (set with the RENGINE= option in the LIBNAME statement) translates the server character representation from ASCII-OEM to EBCDIC for the client.

Export
: To write data to the server, the client REMOTE engine (which is set by default or set with the RENGINE= option in the LIBNAME statement) translates its EBCDIC character representation to ASCII-OEM for the server.

A system administrator uses the TRANTAB= system option to update or replace translation tables or to add new translation tables. For complete details about the TRANTAB= system option, see *SAS Language Reference: Concepts*.

Accessing Version 6 Files or Version 8 Files in a Mixed Library

Note: All Version 8 hosts, except the OS/390 host, support mixed libraries. For a list of Version 8 hosts, see "Host Types Supported According to SAS Release" on page 4. △

You are advised to create version-specific SAS files in version-specific libraries. Such an organization enables you to work on the entire contents of a library in a predictable way. Otherwise, without segregation of files by version, mixed libraries result. A disadvantage of a mixed library is that your access to its members may be limited to files that were created in one version or the other without your awareness. For reasons to convert a Version 6 file to Version 8 format or to access Version 6 and Version 8 files in a mixed library, see "Deciding Whether to Convert the Copied File" on page 13.

Your ability to access a specific file in a library depends on the engine that you associated with the library in the LIBNAME statement. You can use the RENGINE= option in the LIBNAME statement to explicitly invoke the appropriate engine, or you can omit the engine from the LIBNAME statement so that the SAS/SHARE server can determine the appropriate engine to invoke.

Here are examples:

Note: A Version 8 server is assumed in these examples. △

```
libname grades 'SAS-data-library' server=shr1 rengine=V6;
libname grades 'SAS-data-library' server=shr1 rengine=V8;
libname grades 'SAS-data-library' server=shr1;
```

In the first two examples, you explicitly assign to the server library the files that were created in either version. When you specify RENGINE=V6 , the server accesses only files that were created in Version 6. Likewise, when you specify RENGINE=V8, the server accesses only files that were created in Version 8 and Version 7.

Note: Because the Version 7 and Version 8 file formats are identical, the V8 engine will detect both Version 8 and Version 7 files. Likewise, the V7 engine will locate both Version 7 and Version 8 files. △

The advantage of specifying the RENGINE= option is that it overrides the server default, and it reduces search time that is normally required by a server when determining the appropriate engine to invoke. Ordinarily, this option should not be specified because the server determines which engine to use to process the data library.

In the third example, because no 8 server detects a mix of files that have type Version 6 and Version8 formats, then it uses the default V8 engine to access Version 8 (and Version 7) files and it ignores Version 6 files. If the server detects only Version 6 files (names are appended with Version 6 extensions), the server uses the Version 6 engine for file access. Likewise, if it detects only Version 8 (or Version 7) files (names are appended with Version 8 extensions), the server uses the V8 engine for file access.

For example, for a server that is located on an HP-UX UNIX host, the V6 engine permits server access only to data sets whose name extension is **.ssd01**. As another example, for a server on an OpenVMS host, the V8 engine permits server access to data sets whose name extension is **.sas7bdat**. For a complete list of filename extensions by host and by release, see Chapter 17, "SAS Filename Extensions and File Headers," on page 125.

Concatenating Multiple Libraries

In order to expand the scope of file access from a single library to multiple libraries, use library concatenation. With an expanded scope, you can perform operations on files that span multiple libraries. For example, you can concatenate two libraries that were created by the same SAS version or you can concatenate two libraries that were created by different SAS versions.

Here is an example of library concatenation:

```
libname v6lib 'SAS-data-library' server=shr1 rengine=v6;
libname v8lib 'SAS-data-library' server=shr1 rengine=v8;
libname catlib (v8lib v6lib);
```

SAS-data-library must be the physical name that is recognized by the operating environment.

The first LIBNAME statement assigns the libref V6LIB to a SAS data library that is accessed with the V6 remote engine through the SAS/SHARE server SHR1.

The second LIBNAME statement assigns the libref V8LIB to a SAS data library that is accessed with the V8 remote engine through the SAS/SHARE server SHR1.

The third LIBNAME statement assigns the libref CATLIB to concatenated libraries that are referenced by the librefs V8LIB and V6LIB. The order of the librefs identifies the sequence in which the libraries are searched. The SAS operation uses the first occurrence of the specified file.

For example, if the same file exists in both libraries and you delete that file, then the file in the first library (for example, STOCK.SAS7BDAT in V8LIB) is deleted. If V6LIB precedes V8LIB in the library concatenation (for example, STOCK.SD2 in V6LIB), then that file is deleted. If the specified file exists only in one library, then that file is deleted. For a complete list of filename extensions according to member type and version, see .“Using Filename Extensions to Determine Which SAS Engine and Host Were Used to Create a SAS File” on page 125

Accessing Files in a Cross-Version Environment

Consider a cross-version environment in which a SAS/SHARE client accesses files on a SAS/SHARE server, with each running a different SAS version from the other. These factors determine your ability to access (read, write, or update) a file in such an environment:

□ SAS version run by the SAS/SHARE server

□ SAS version run by the SAS/SHARE client

□ SAS version of the file being accessed

□ Member being accessed.

Accessing Data Sets in a Cross-Version Environment

Table 7.2 on page 62 summarizes the permissions to access data sets in a cross-version environment.

Table 7.2 Accessing Data Sets in a Cross-Version Environment

	V6 Server		**V8 or V7 Server**	
	V6 Data Set	**V7 or V8 Data Set**	**V6 Data Set**	**V8 or V7 Data Set**
V6 Client	RWU	No Access	RWU	RWU[1]
V8 or V7 Client	RWU[1]	No Access	RWU[1]	RWU

RWU means that read, write, and update access is supported. A Version 6 client’s ability to recognize Version 7 or Version 8 files on a Version 7 server or Version 8 server is determined by the server’s base engine.

No Access means that neither read, write, nor update access is supported.

[1] indicates that features that are specific to Version 7 and Version 8 , such as long names, are not supported.

Accessing DATA Step and PROC SQL Views in a Cross-Version Environment

Table 7.3 on page 63 summarizes the permissions to access DATA step views and PROC SQL views in a cross-version environment.

Table 7.3 Accessing DATA Step Views and PROC SQL Views in a Cross-Version Environment

	V6 Server		**V8 or V7 Server**	
	V6 View	**V8 or V7 View**	**V6 View**	**V8 or V7 View**
V6 Client	RWU	No Access	RWU[1]	R[2]
V8 or V7 Client	R[2]	No Access	R[3]	RWU

In this table, View refers to DATA step views and PROC SQL views only.
RWU means read, write, and update access is supported.
No Access means that neither read, write, nor update access is supported.

[1] At either the client or the server, the RMTVIEW=NO option and the argument to the LIBNAME statement has been set. For more information about this option, see "Deciding Where Views Should Be Interpreted" on page 58.

[2] At either the client or the server, the RMTVIEW=YES option and the argument to the LIBNAME statement has been set. For more information about this option, see "Deciding Where Views Should Be Interpreted" on page 58.

[3] Applies to PROC SQL views only.

Accessing SAS/ACCESS Views in a Cross-Version Environment

Table 7.4 on page 63 summarizes the permissions to access SAS/ACCESS views in a cross-version environment.

Table 7.4 Accessing SAS/ACCESS Views in a Cross-Version Environment

	V6 Server		**V8 or V7 Server**	
	V6 Data Set	**V8 or V7 Data Set**	**V6 Data Set**	**V8 or V7 Data Set**
V6 Client	RWU	No Access	RWU	RWU
V8 or V7 Client	RWU	No Access	RWU	RWU

RWU means that read, write, and update access is supported.
No Access means that neither read, write, nor update access is supported.

Accessing Catalogs in a Cross-Version Environment

Table 7.5 on page 64 summarizes the permissions to access catalogs in a cross-version environment.

Table 7.5 Accessing Catalogs in a Cross-Version Environment

	V6 Server		**V8 or V7 Server**	
	V6 Catalog	**V8 or V7 Catalog**	**V6 Catalog**	**V8 or V7 Catalog**
V6 Client	RWU	No Access	R	RWU[1]
V8 or V7 Client	RW[2]	No Access	RW[2]	RWU

RWU means that read, write, and update access is supported.

No Access means that neither read, write, nor update access is supported.

[1] indicates that this update operation produces a Version 8 or Version 7 catalog that contains Version 6 data.

[2] indicates that this write operation allows only a limited copy. You can use PROC COPY to copy a V6 catalog to another V6 library.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *Moving and Accessing SAS Files across Operating Environments, Version 8*, Cary, NC: SAS Institute Inc., 1999. 186 pages.

Moving and Accessing SAS Files across Operating Environments, Version 8

ISBN 1-58025-480-2

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

1st printing, October 1999

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