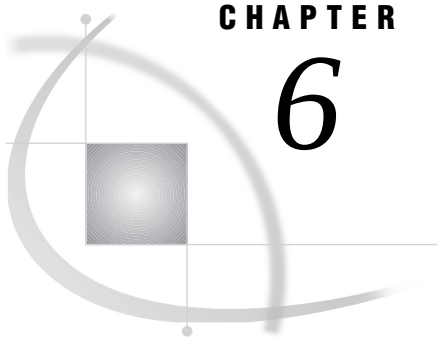

CHAPTER

# 6

# Using SAS/SHARE Macros for Server Access



## Introduction

This chapter describes how server administrators can use macros to create a flexible and dynamic server environment. See Chapter 14, “SAS/SHARE Macros,” on page 133 for complete details about each macro that you can use to access a server and its libraries.

## Audience

This chapter is recommended primarily for server administrators and programmers who write applications that access shared data.

## Why Use Macros for Server Library Access?

Programs that use SAS/SHARE software must include a LIBNAME statement to identify the SAS/SHARE server through which a library is to be accessed. Adding servers and changing serverids can require users and server administrators to obtain

current serverid information each time they want to access a server. Thus, maintaining production or utility programs can be very difficult.

Moreover, although there is no permanent connection between a SAS library and a given server, there is frequently a logical one. Programmers, server administrators, and users may always want a library to be accessed through the same server, particularly because only one server can provide access to a library at a time. The same logical connection can also exist between a group of users and a server. It may be desirable or necessary for all the members of a department to use the same server, especially if they are sharing libraries.

To make the most of SAS/SHARE software without compromising performance, administrators often need a dynamic and flexible server environment. They need to be able to start and stop servers as the need arises, and to easily redistribute the load on the servers. They want to be able to switch libraries and users from one server to another quickly and easily. To balance the needs of both administrators and users, SAS/SHARE software includes macros to be defined through the autocall function of the SAS macro facility.

These SAS/SHARE macros enable the administrator to define aliases for a server and to associate an alias with a given library. Programs can then issue these macros to generate the requisite LIBNAME statements that access the library through the server that is associated with the alias. Thus, the administrator can add servers, change serverids, and switch libraries and users from one server to another in a manner that is totally transparent to the program or SAS user.

There are macros to do the following tasks:

- generate and display the tables of macro variables that associate libraries with server aliases and server aliases with serverids
- generate part or all of a LIBNAME statement
- start and stop servers
- generate PROC OPERATE and SET SERVER statements.

The serverid that is associated with an alias can be changed during any appropriate server or application outage (for example, down time). You update only the file that contains the table of macro variables that maps aliases to serverids. Additionally, a library can be logically associated with a different server by updating the table that associates libraries with server aliases.

For example, a site might have four logical servers (that is, four different server aliases) but only one physical server by having all the aliases map to the same serverid. Whenever the load on that single server gets too heavy, the site can start an additional server and shift certain libraries and users to it by simply pointing one of the aliases to that new server.

## The SAS/SHARE Macros

SAS/SHARE software includes three categories of autocall macros:

- utility macros that can be used by all SAS/SHARE programs and sessions
- macros used in user programs
- macros used in server administrator programs.

### Utility Macros

The utility macros and their functions are:

SHRMACS
: compiles all the other macros and builds the server-alias and library-alias tables.

SERVERID
takes a server alias and looks up the serverid in the server-alias table and generates *serverid* or SERVER=*serverid*, as appropriate.

SERVIIDX
returns the index of the entry in the server information table for the specified server.

LISTSRV
writes the server-alias table to the log.

LISTLIB
writes the library-alias table to the log.

LISTSRVI
lists the server information table to the log.

### User Program Macro

The user program macro and its function is

LIBDEF
takes a libref and an optional physical library name and looks up the SAS library name in the library-alias table and generates a LIBNAME statement.

### Server Administrator (Operator) Macros

The macros used in server administrator programs are:

STRTSRV
starts a server with the appropriate serverid by using the SERVERID macro to convert the alias. The macro takes a server alias and PROC SERVER statement options.

SHUTSRV
generates the PROC OPERATE statement and STOP SERVER command for the appropriate serverid by using the SERVERID macro to convert the alias. The macro takes a server alias and an optional password.

OPERATE
generates PROC OPERATE statements for the appropriate serverid by using the SERVERID macro to convert the alias. The macro takes a server alias and an optional password.

SETSRV
generates a SET SERVER serverid statement by using the SERVERID macro to convert the alias. The macro takes a server alias and an optional password.

## Other SAS/SHARE Macros Generated by the SHRMACS Macro

### Compiling Macros (SHRMACS)

A server administrator or an applications programmer must invoke the SHRMACS macro to compile all other macros. You always invoke the SHRMACS macro before any other macro. Use the following syntax:

%SHRMACS(*category*,<*log-info*,>

<APPLSYS=*app-sys-lib-tab*,>
<SASSAML=*alt-sys-lib-tab*>);

where

*category*
specifies the category of macros to be compiled (SERVER, USER, OPER, or ALL). See Table 6.1 on page 78.

*log-info*
indicates whether descriptive information is written to the SAS log about each macro (NOMSG, MSG, or HELP). The default is MSG.

APPLSYS=
enables you to specify an alternate applications systems library-alias table. For details about specifying the APPLSYS= argument, see "Setting Up the Application System (APPLSYS) Macro Library" on page 80.

SASSAML=
enables you to specify an applications systems library, which is a set of files that specify SAS data libraries and servers. For details about specifying the SASSAML= argument, see "Setting Up the Application System (APPLSYS) Macro Library" on page 80.

Besides compiling the requested macros, the SHRMACS macro also builds the appropriate library-alias table and server-alias table. These tables are used for generating the server name for the PROC SERVER, PROC OPERATE and LIBNAME statements. Table 6.1 on page 78 lists the categories of macros that can be compiled and the implicit macros that are generated, by category.

**Table 6.1** Other SAS/SHARE Macros Generated by the SHRMACS Macro

| **%SHRMACS(*category*)** | | | |
|---|---|---|---|
| Server | User | Operator | **Macros Generated** |
| • | • | • | SERVERID |
| • | • | • | LISTLIB |
| • | • | • | LISTSRV |
| • | | | STRTSRV |
| | • | | LIBDEF |
| | | • | SHUTSRV |
| | | • | OPERATE |
| | | • | SETSRV |
| • | • | • | SERVIIDX |
| • | • | • | LISTSRVI |

For example, all of these macro definitions

%SHRMACS(SERVER)

%SHRMACS(USER)

%SHRMACS(OPER)

generate these macros: SERVERID, LISTLIB, LISTSRV, STRTSRV, LISTSRVI, and SERVIIDX.

However, only %SHRMACS(USER) generates the LIBDEF macro, only %SHRMACS(SERVER) generates the STRTSRV macro, and only %SHRMACS(OPER) generates the SHUTSRV, OPERATE, and SETSRV macros.

The SERVER category of macros generates the server-alias table; the OPER category of macros generates the library-alias table; and the USER category of macros generates both the server-alias and the library-alias tables.

Specifying the ALL category of macros generates all of the macros as well as both the server-alias and library-alias tables.

Output 6.1 on page 79 shows information about the server macros, as listed in the SAS log.

**Output 6.1** Server Macro Information

```
LOG
 Command ===>
1085  %shrmacs (server,msg);

            *** SAS/SHARE macros are now available ***

For further information about SAS/SHARE macros:

   %SHRMACS(ALL,HELP)    - for information on all macros
   %SHRMACS(USER,HELP)   - for information on macros used
                           in user applications
   %SHRMACS(OPER,HELP)   - for information on macros used
                           with PROC OPERATE
   %SHRMACS(SERVER,HELP) - for information on macros used
                           with PROC SERVER
   or %macro(HELP)       - for information on a specific
                           macro

SAS/SHARE macros generated are:

SERVERID - translate server alias
LISTLIB  - list library table
LISTSRV  - list server alias table
STRTSRV  - start a server

                        SERVER ALIAS TABLE

--- SERVER ALIAS ------- SERVERID ----------------------------
    CESERV               V6DSERVR
    COMSERV              MYSERV
    DEVSERV              MYSERV
    GLOSSERV             V6DSERVR
    LIBSERV              V6DSERVR
    MISSERV              MYSERV
    PRDSERV              V6DSERVR
-------------------------------------------------------------
```

For more information about the macros that are compiled by SHRMACS, you can specify the HELP keyword, as shown in this example:

```
%shrmacs(server,help);
```

HELP lists the syntax, a brief description, and an example for each macro that SHRMACS defines in the SERVER category.

# Setting Up the Application System (APPLSYS) Macro Library

The data that is used by the macros is stored in tables in another macro library that you maintain as you add and delete libraries, servers, and application systems. This library is called the *APPLSYS* (an acronym for application system) *macro library*.

The following tasks to customize macros are explained:

- specifying the APPLSYS macro library
- defining server aliases (SERVID)
- associating SAS libraries with servers (SERVLIB)
- setting up a server information table (SERVINFO)
- generating a LIBNAME statement (LIBDEF)
- using the APPLSYS argument in the SHRMACS and the LIBDEF macros.

See Chapter 14, "SAS/SHARE Macros," on page 133 for complete information about the syntax of the SAS/SHARE autocall macro library and how to use it.

## Specifying the APPLSYS Macro Library

The default name of the APPLSYS macro library is host-dependent. The default macro library names follow:

CMS
SASSAML MACLIB

OpenVMS
SAS$ROOT:[SASSAML]

OS/2
*!SASROOT*\SHARE\SASMACRO

OS/390
SAS.SASSAML

UNIX
***!sasroot*/saspgm/sassaml**

Windows
*!SASROOT*\SHARE\SASMACRO

Files in the APPLSYS macro library must have a .SAS extension. These files are referred to as members.

To use the default library table, you omit the APPLSYS= argument to the SHRMACS macro. For example,

```
%shrmacs(user);
```

To specify an alternate library-alias table, supply the APPLSYS= argument to the SHRMACS macro. For example:

```
%shrmacs(user, applsys=purchas);
```

It may be more convenient to allow different users or departments to maintain their own APPLSYS macro libraries. You can indicate that an alternate APPLSYS macro library be used instead of the default library by specifying the SASSAML= argument to the SHRMACS macro.

The value of this argument can be the host-specific physical name of the alternate library or the string _DEFINED_, which indicates that the fileref SASSAML has already been assigned to the alternate library. For example:

```
%shrmacs(user,applsys=purchas,sassaml=library-path);
```

or

```
%shrmacs(user,applsys=_DEFINED_);
```

Typical host-specific examples of how to specify an alternate APPLSYS macro library follow:

OpenVMS

```
%shrmacs(user,applsys,sassaml=MIS$:[applsys]);
```

OS/2

```
%shrmacs(user,applsys,sassaml=c:\dept\mis\applsys);
```

UNIX

```
%shrmacs(user,applsys,sassaml=/dept/mis/applsys);
```

Windows

```
%shrmacs(user,applsys,sassaml=c:\dept\mis\applsys);
```

## Defining Server Aliases in the Server-Alias Table (SERVID)

Server aliases can be helpful for

- using your existing SAS programs with new releases of SAS/SHARE software without changing them. This is accomplished by letting the name of the existing SAS server be an alias for the name of a SAS server that is executing the new release of SAS/SHARE software.
- shifting server traffic easily. Early in your use of SAS/SHARE software, you might create many aliases for a single server, where each alias is used by only one or a few of your applications. As server use increases, you can add a second server and switch some of your applications to that server. You do this by simply changing the entry in the SERVERID member for those applications to point to your new server.

To define server aliases, create a member named SERVERID in the APPLSYS macro library. The member name must be SERVERID because the SHRMACS macro looks for that specific name.

Define a server alias in the SERVERID member using the following syntax:

%SERVID(*alias,serverid*);

Example Code 6.1 on page 81 shows an example of a SERVERID member.

**Example Code 6.1** Server-Alias Table

```
/************************************************/
/*                                              */
/*  NAME: SERVERID                              */
/*                                              */
/*  SERVER ALIAS TABLE ENTRIES                  */
/*                                              */
/*  This member defines aliases for server names.  */
/*  The entries in this member are loaded into     */
/*  the server alias table by the SHRMACS macro.   */
/*  This table is used by the SERVERID macro to    */
/*  translate an alias to an actual serverid.      */
/*                                              */
/*  To add aliases to the table, add a SERVID      */
/*  call for each alias at the end of this         */
```

```
/*  member. Specify the alias and then the         */
/*  real serverid.                                 */
/*                                                 */
/*************************************************/
%servid(shr7,shrserv7)
%servid(share1,shrserv3)
%servid(pubserv,shrserv7)
```

To add aliases to the table, use a SERVID call for each alias-serverid pair.

## Associating SAS Libraries with Server Aliases (SERVLIB)

Create a member in the APPLSYS macro library for each application system that is specified by the APPLSYS= argument to the SHRMACS macro. Doing this defines library-server pairs that a specific application is likely to use. For example, if you specify an APPLSYS library named PURCH in the SHRMACS macro as follows,

```
%SHRMACS(user,APPLSYS=purch);
```

you also create a member in the APPLSYS library of the same name in this example, PURCH.

In a selected member, specify the library and server name pairs by using the following syntax:

%SERVLIB(*SAS-data-library, server-name*);

where *SAS-data-library* is specific to the host, and *server-name* can be represented as a serverid or its alias.

Example Code 6.2 on page 82 shows an example of a member named PURCH, which contains references to host-specific SAS library names.

**Example Code 6.2** Library-Alias Table

```
/*******************************************************************/
/*                                                                 */
/*  NAME: PURCH                                                    */
/*                                                                 */
/*  LIBRARY TABLE ENTRIES - SPECIFIC APPLICATION                   */
/*                                                                 */
/*  This member associates server names with libraries. The entries   */
/*     in this member are loaded into the library table if the        */
/*     SHRMACS macro is called by using the argument APPLSYS=APPLSAMP.*/
/*     The entries can also be loaded by using a call to the LIBDEF   */
/*     macro if APPLSYS=APPLSAMP is specified.                        */
/*                                                                 */
/*  To add libraries to the definition table, add a SERVLIB call      */
/*    for each library at the end of this member. Specify the         */
/*    physical name for the library and the name of the server to be  */
/*    associated with the library. The name may be an alias or an     */
/*    actual serverid.                                               */
/*                                                                 */
/*******************************************************************/
%servlib(appljan c, testserv);                          # CMS
%servlib(disk1$:[shrtest.appljan.lib1), testserv);      # OpenVMS
%servlib(d:\shrtest\appljan\lib1, testserv);            # OS/2
%servlib(shrtest.appljan.lib1, testserv);               # OS/390
%servlib(/shrtest/appljan/lib1, testserv);              # UNIX
```

```
%servlib(d:\shrtest\appljan\lib1, testserv);              # Windows
```

To add aliases to the table, use a SERVLIB call for each library-server pair.

Additionally, create a member in the APPLSYS macro library named DEFAULTS. This member can be empty, but must be present to avoid error messages from the SHRMACS macro.

The DEFAULTS member is used when the APPLSYS= argument is omitted on a call to the SHRMACS and LIBDEF macros.

The syntax of the PURCH and the DEFAULTS members are identical.

## Creating the Server Information Table and Adding Server Attributes (SERVINFO)

You create a server information table to store information about the servers at your site. You can use this information in a program or you can display it. By default, the table contains the following information:

- a default value for the REMOTE engine's LIBNAME statement option RMTVIEW=
- a network node name that is represented in a two-level server name format: *node.server-id*.

You can use this table to store other attributes of the server or its users or administrators, such as server access passwords, PROC SERVER statement options, and the release of SAS software that the server runs under.

Use the SERVINFO macro to add an entry to the server information table. Use the following syntax:

%SERVINFO (*node.server-id*, netnode=*fully-qualified-node-name*, RMTVIEW=NO);

A typical use of this macro is for the SERVERID macro to generate an alias for a node name that is not a valid SAS name. For example, when the two-level server name and the netnode are specified in the server information table as follows:

```
%servinfo (hp.shrserv,netnode=hp103.dom2.acme.com);
```

server SHRSERV runs on HP103.DOM2.ACME.COM.

In resolving an alias for HP.SHRSERV, SERVERID generates

```
%let hp=hp103.dom2.acme.com;
```

## Customizing a Server Information Table

This example shows how to customize your server information table to include the SAS/SHARE server's SAS software release level. The steps in the example accomplish the following objectives:

- account for a new parameter in the SERVINFO macro
- reformat the table's appearance
- add the new information to the table
- view the server information table with the LISTSRVI macro.

**1** Add new parameter SASREL= to the SERVINFO macro statement, as follows:

```
%macro servinfo(servid,version=,rmtview=,
netnode=,sasrel=);
```

**2** Add a new variable ISREL&SRVINUM to the %GLOBAL statement to account for the new parameter SASREL, as follows:

```
%GLOBAL isrvr&srvinum irmtv&srvinum
   inode&srvinum isrel&srvinum;
```

3 Because the table is implemented as sets of macro variables, assign the value &SASREL to the macro variable ISREL&SRVINUM as follows:

```
%LET isrel&srvinum = &sasrel;
```

4 In the LISTSRVI macro, modify the line that prints the headers for the table as follows:

```
%put &pline RMTVIEW %shrrpt(-,3)
   NETWORK NODE %shrrpt(-,20) RELEASE %shrrpt(-,3);
```

5 For aesthetics, you may want to change the %PUT statement in the LISTSRVI macro to extend the dashed line following the table so that it matches the length of the modified header line.

```
%put %shrrpt(-,78);
```

6 Change the loop that prints the table so that it looks like this:

```
%do i=1 %to &srvinum;
  %let pline=%shrrpt(&blank,3)
     %shrfmt(&&isrvr&i,16);
  %let pline=&pline
     %shrfmt(&&irmtv&i,11)
     %shrfmt(&&inode&i,36);
  %let pline=&pline &&isrel&i;
  %put &pline;
%end;
```

The first six steps illustrate how you would alter the server information table for display only.

7 If you wanted to be able to access and use the information in the table for a macro or a program, you would have to use the SERVIIDX macro, as shown in the following code fragment:

```
%let i=%serviidx(&new_id);
%if (&&isrel&i^=) %then
  %do;
   /* some use of &&isrel&i here */
  %end;
```

8 After you have accounted for the new parameter and modified the format of the server information table, you will actually add entries to the table for all parameters as follows:

```
%servinfo(rmthost.share1,netnode=.acme.com,
  rmtview=no,sasrel=6.12);
%servinfo(rmthost.share2,netnode=smith.acme.com,
  rmtview=yes,sasrel=7);
```

The following code shows the server information table that is sent to the SAS log when you invoke the LISTSRVI macro.

```
%listsrvi;
               SERVER INFORMATION TABLE
--- SERVERID --- RMTVIEW -- NETWORK NODE --- RELEASE
RMTHOST.SHARE1   NO        rmthost.acme.com   6.12
RMTHOST.SHARE2   YES       smith.acme.com     7
-------------------------------------------------
```

## Generating a LIBNAME Statement (LIBDEF)

If your SAS application accesses a server library, use the LIBDEF macro instead of a LIBNAME statement. Use the following syntax:

**%LIBDEF**(*libref,SAS-data-library-name*) <,APPLSYS=*app-sys-lib-tab*>;

*Note:* Do not enclose the SAS data library name in quotation marks. Using quotes will cause the generation of the LIBNAME statement to fail. △

*Note:* Before you invoke the LIBDEF macro, you must first invoke the SHRMACS macro. △

The LIBDEF macro generates a LIBNAME statement by searching the library table for the library name. It then invokes the SERVERID macro to convert the server alias into a serverid.

Host-specific examples of how to use the LIBDEF macro follow:

CMS

```
%libdef(mylib,appljan c);
```

OpenVMS

```
%libdef(mylib,disk1$:[shrtest.appljan.lib1);
```

OS/2

```
%libdef(mylib,d:\shrtest\appljan\lib1);
```

OS/390

```
%libdef(mylib,shrtest.appljan.lib1);
```

UNIX

```
%libdef(mylib/shrtest/appljan/lib1);
```

Windows

```
%libdef(mylib,d:\shrtest\appljan\lib1);
```

Each macro invocation defines the library to the libref MYLIB. You can define additional libraries without invoking the SHRMACS macro again.

If you want to use server aliases but you have not created the library- and server-name pairs in the APPLSYS macro library yet, you can use a LIBNAME statement and invoke the SERVERID macro in place of the SERVER= argument. See Chapter 14, “SAS/SHARE Macros,” on page 133 for more information.

## Using the APPLSYS= Argument in the SHRMACS and LIBDEF Macros

You can specify the APPLSYS= argument in either the SHRMACS or the LIBDEF macro. Example Code 6.3 on page 86 illustrates application excerpts that show how to access libraries from three different applications systems (PURCH, MAINT, and FACIL) that use the APPLSYS= argument to the SHRMACS and LIBDEF macros, as appropriate.

**Example Code 6.3** Calling the SHRMACS and LIBDEF Macros with the APPLSYS= Argument

```
/* Most libraries will come from purchasing appl sys.*/
%shrmacs(user,nomsg,applsys=purch);

/* Access purchase order library.*/
%libdef(polib,SAS-data-library1);
            .
            .
            .
            .
/* Access vendor service library from maintenance appl sys.*/
%libdef(vndsvc,SAS-data-library2,applsys=maint);
            .
            .
            .
            .
/* Access vendor account library from purchasing appl sys. */
/*                                                         */
                                                           */
/* (Note: It is not necessary to specify APPLSYS= for this */
/* application system because it was specified above.)     */
/*                                                         */
%libdef(vndacct,SAS-data-library3);


/* Access vendor contact library from maintenance appl sys.*/
/*                                                         */
/* (Note: It is not necessary to specify APPLSYS= for this */
/* application system because it was specified above.)     */
/*                                                         */
%libdef(vndcon,SAS-data-library4);
            .
            .
            .
            .
/* Access inventory library from facilities appl sys.      */
%libdef(invlib,SAS-data-library5,applsys=facil);

/* Access invoice library from purchasing appl sys.        */
%libdef(invoice,SAS-data-library6);
```

*Note:* You supply a *SAS-data-library* using the syntax convention that is appropriate for your host. △

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/SHARE User's Guide, Version 8*, Cary, NC: SAS Institute Inc., 1999. pp. 247.

**SAS/SHARE User's Guide, Version 8**



ISBN 1–58025–478–0



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

1st printing, September 1999