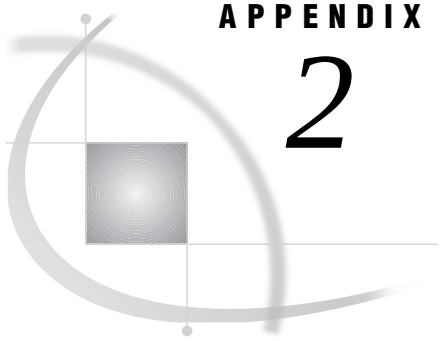

APPENDIX

2

Creating the SAS/SHARE Server Environment

Introduction

Before you start a SAS/SHARE server on your host, you may need to prepare your operating environment to accommodate it. Requirements for successful server start-up and operation vary according to host type. This appendix describes site requirements by host.

Audience

This appendix is designed for system administrators or server administrators who are responsible for preparing the operating environment to accommodate a SAS/SHARE server.

All Hosts: Setting SAS System Performance Options

Several SAS system options can help you reduce the number of disk accesses needed for SAS files and, therefore, enhance system performance. The SAS system options listed here are valid for all operating environments. In addition, options that are host specific are documented in the subsequent host-specific sections.

Some of the options also have host-specific values, which are described in the "Setting SAS System Performance Options" section for each operating environment.

From a SAS session, run PROC OPTIONS to find the default settings for SAS system options on your system.

BUFNO=*n* | *n*K | *n*M | *n*G | MAX | MIN | *hex*X
specifies the number of buffers to use for SAS data sets. The number of buffers is not a permanent attribute of the data set, and it is valid only for the current SAS session or job. The BUFNO= option applies to SAS data sets opened for input, output, or update. If the number of buffers is 0, SAS uses the operating environment default values.

Using the BUFNO= option can speed up execution time by limiting the number of I/O operations required for a particular SAS data set. The improvement in execution time, however, comes at the expense of increased memory consumption. For SAS/SHARE, setting the BUFNO= option too high may hurt performance by using too much memory because SAS/SHARE may be accessing multiple files at once.

You can use the following values with the BUFNO= option:

n | *n*K | *n*M | *n*G
specifies the number of buffers in the form of *n multiplier*, where the value of each multiplier is 1; 1,024; 1,048,576; and 1,073,741,824, respectively. For example, a value of 8 specifies 8 buffers; a value of 4K specifies 4,096 buffers; and a value of 3M specifies 3,145,728 buffers.

MAX
sets the number of buffers to the largest signed, 4-byte integer that can be represented in your operating environment.

MIN
sets the number of buffers to 0, and requires SAS to use a default value.

*hex*X
specifies the number of buffers as a hexadecimal number that must be followed by an X.

BUFSIZE=*n* | *n*K | *n*M | *n*G | MAX | MIN | *hex*X
specifies the permanent buffer size for an output SAS data set. If the number of bytes is greater than 0 when a SAS data set is created, that number is used as the default value for the BUFSIZE= data set option. If the BUFSIZE= data set option is omitted and the number of bytes for the BUFSIZE= system option is 0, SAS

chooses an operating environment default value that is optimal for the SAS data set.

Using the BUFSIZE= option can speed up execution time by limiting the number of I/O operations required for a particular SAS data set. However, the improvement in execution time comes at the expense of increased memory consumption. For SAS/SHARE, setting the BUFSIZE= option too high may hurt performance by using too much memory because SAS/SHARE may be accessing multiple files at once.

You may want to vary the value of the BUFSIZE= option if you are trying to maximize memory usage or the number of observations per page.

n | *n*K | *n*M | *n*G
specifies the permanent buffer size in bytes, kilobytes, megabytes, or gigabytes, respectively.

CATCACHE=
specifies the number of SAS catalogs to keep open. Use the CATCACHE= option to tune an application by avoiding the overhead of repeatedly opening and closing the same SAS catalogs. Typically, you should leave this option set to 0. Increasing its value can use up memory.

You may use the following values for the CATCACHE= option:

n | *n*K | *n*M | *n*G
specifies any integer greater than or equal to 0 in bytes, kilobytes, megabytes, or gigabytes, respectively. If *n*>0, SAS places up to that number of open-file descriptors in cache memory instead of closing the catalogs.

MAX
sets the number of open-file descriptors kept in cache memory to the largest signed, 4-byte integer that can be represented in your operating environment.

MIN
sets the number of open-file descriptors kept in cache memory to 0.

*hex*X
specifies the number of open-file descriptors kept in cache memory as a hexadecimal number. This number must be followed by an X.

COMPRESS=YES | NO
controls the compression of observations in output SAS data sets. Compressing a data set reduces the size of the data set by reducing repeated consecutive characters to two- or three-byte representations. Compression of observations is not supported by all engines. To uncompress observations, use a DATA step to copy data and specify COMPRESS=NO for the new data set.

The advantages gained by using the COMPRESS= option include:

- reduced storage requirements for the data set
- fewer I/O operations necessary to read from or write to the data set during processing

You can use the following values with the COMPRESS= option:

YES
specifies that observations in a newly created SAS output data set are compressed (variable length records).

NO
specifies that observations in a newly created SAS output data set are uncompressed (fixed length records).

A typical SAS configuration file follows:

```
bufno=1
bufsize=65024
catcache=0
compress=YES
```

See *SAS Language Reference: Dictionary* for more information about these SAS system options.

CMS: Creating the Server Environment

You must perform the following steps to create the server environment on a CMS host:

1. Set SAS system performance options.
2. Create a VM directory entry for the server virtual machine.
3. Format the server minidisks.
4. Create a PROFILE EXEC for the server.
5. Place SAS libraries on the server minidisks.

Setting SAS System Performance Options

The following SAS system options can be used to tune server performance.

BLKSIZE=*number-of-bytes*
specifies the size of a contiguous buffer for each open SAS file. The SAS default option BLKSIZE= is set to enable optimum I/O buffering during sequential processing of SAS data libraries. Because a server's processing is primarily random, the default value of this option can actually *increase* the amount of I/O that is performed within a server and can cause the performance of the server to be less efficient. The value of the BLKSIZE= option increases the amount of data that is read or written when a user requests that I/O operations be performed on the shared data. The performance of a server is less efficient when the increased amount of data is of no relevance to the individual user's request for specific portions of the data.

Specifying BLKSIZE=0 when invoking SAS in the server environment can help avoid unnecessary use of the server's virtual storage and can help you reduce the amount of I/O performed in the server execution.

MEMSIZE=*n* | *n*K | *n*M
specifies a limit on the total amount of memory that SAS uses at any one time. The server should be allowed a large amount of memory in order to accommodate the needs of all its users. The number of concurrent SAS files open by server users and the number and size of I/O buffers for each SAS file have a direct impact on the amount of memory required by the server. You should be aware that, in some cases, SAS terminates abnormally when it is unable to satisfy a memory request.

You can use the following values with the MEMSIZE= option:

n
specifies a number in the range 0 to 2,147,483,648. A value of 0 indicates that there is no limit except the operating environment limit, which may be site specific.

*n*K
specifies the number of kilobytes, from 0 to 2,097,152.

*n*M
specifies the number of megabytes, from 0 to 2048.

For the CMS environment, the default is MEMSIZE=0. In order to limit the amount of virtual memory that SAS can use, you should specify a value for the MEMSIZE= option. For SAS/SHARE, you may want to specify a value of 16M or 20M for the MEMSIZE= option to allow for peak usage periods.

VIOBUF=*number-of-bytes*
specifies the size of the virtual I/O buffer to be used for WORK data sets. Because a server does not use data sets in the WORK library, VIO memory is not used by a server.

Specifying a value of 0 (zero) for this option allows a server to make more efficient use of its virtual storage.

UNBUFLOG
specifies that the SAS log file (which was specified in the ALTLOG= option) is opened so that other processes can read it, and that each line written to the log is then immediately transferred to disk. This option enables you to examine the server SAS log while the server is running. The UNBUFLOG option must be specified in a SAS command or in a SAS configuration file.

Note: The overhead that is incurred by the UNBUFLOG option may degrade the performance of a busy server. △

A typical SAS configuration file follows:

```
blksize=0
memsize=20M
viobuf=0
unbuflog
```

See *SAS Companion for the CMS Environment* for more information about these SAS system options.

Creating a VM Directory Entry for the Server Virtual Machine

To use SAS/SHARE software, you must first create a VM directory for each server virtual machine. The following code example shows a typical VM directory entry for a server virtual machine.

Example Code A2.1

```
USER SASSHARE XXXXXXX 20M 20M G 100
MCHINE XA
IPL CMSXA PARM AUTOCR
OPTION MAXCONN 1024 REALTIMER
IUCV ALLOW PRIORITY MSGLIMIT 255
CONSOLE 009 3215
SPOOL   00C 2540 READER 8
SPOOL   00D 2540 PUNCH A
SPOOL   OOE 1403 A
LINK MAINT 19E 19E RR
LINK MAINT 19D 19D RR
LINK MAINT 190 190 RR
MDISK 191 3380 707 5 VM0800 MR XXXXXXX
```

```
MDISK 192 3380 501 3 VM0450 MR XXXXXXX
```

The following section describes the directory records for the VM directory entry for a server virtual machine.

USER Directory Record

The default virtual storage size and the maximum virtual storage size of the server virtual machine should be the same. It is impractical to stop the server, define its virtual machine storage size to a large value, and then start the server again.

The amount of virtual storage that a server requires depends on the following factors:

- whether the SAS supervisor and base procedures are installed in saved segments that are not contiguous (discontiguous saved segments-DCSS)
- the number of users who simultaneously access the server
- the internal data set pagesize of the data sets that are shared through the server (as reported by the CONTENTS procedure)
- the number of SAS files that each user accesses at one time through the server.

The preceding VM directory entry code example shows that the server executes in a 20-megabyte virtual machine, but 20 megabytes may not be the optimal virtual storage size for the SAS server at your site. Use the default storage size for interactive virtual machines at your site, and then alter (increase or decrease) the value if necessary.

If you run a VM/XA operating system and want to provide your server with sufficient virtual storage, you may use one or more XA-mode virtual machines.

A server does not require a CP command privilege class that is higher than G.

The dispatching priority of the server virtual machine should initially be the default value for interactive VM users. Monitor your system's performance to determine whether you should assign a higher or lower dispatching priority.

IPL Directory Record

The server virtual machine should be IPL CMS (or CMSXA). Do not specify PARM AUTOCR if the class A or B CP command AUTOLOG is used to log on to the server because the AUTOLOG command passes a line of data to satisfy the server virtual machine's first console read. However, you should specify PARM AUTOCR when running the server interactively. No negative effects have been observed when PARM AUTOCR is specified and an AUTOLOG command is used to log on to the server virtual machine.

OPTION Directory Record

Specify the REALTIMER option while defining the server virtual machine. The effect of the REALTIMER directory option can be duplicated by including the CP command SET TIMER REAL in the PROFILE EXEC of the server virtual machine.

The MAXCONN parameter of the OPTION record specifies the maximum number of SAS users under CMS who can connect to the server at one time. IBM limits the value that you can specify for MAXCONN to 65535. SAS/SHARE software does not specifically limit the number of simultaneous connections to a server.

IUCV Directory Record

The IUCV directory record is required for the server's virtual machine.

The ALLOW parameter enables users to establish IUCV connections to the server virtual machine.

The PRIORITY parameter enables the server to send IUCV priority messages to users.

The MSGLIMIT parameter increases the number of outstanding messages that are allowed on each path from 10 (the default) to 25.

MDISK Directory Records

Because the modules that constitute SAS/SHARE software on CMS reside either on your SAS System or on the SAS/SHARE minidisk, the server requires no minidisk space of its own for program storage.

When you configure your first server, you may want to use the single read-write minidisk at virtual address 191, file mode A. The minidisk should be large enough to hold all the SAS data sets that are shared through the server. If you already have SAS data sets that you copy to the server's minidisk, the minidisk must be large enough to contain all the data sets. You should also allow some extra room for growth.

Alternatively, you can allocate several minidisks to a server. This enables you to divide space according to departments and application systems.

If a DATA step or a procedure is used to replace server data sets, you must allow space for two copies of each data set that the DATA step or the procedure replaces. You rarely need to allow space for all server data sets on a minidisk to be replaced at the same time. Instead, find the DATA step or the procedure that replaces the largest server data sets, and allow room for two simultaneous copies of those data sets.

Formatting the Server Minidisks

At some sites, the VM system administrator formats minidisks when a new virtual machine is defined. However, if you must format the server minidisks, you should specify a BLKSIZE= option of 4K or 4096 in the CMS FORMAT command. This allows the maximum amount of data to be transferred in each I/O operation, which is the most efficient for the server.

Creating a PROFILE EXEC for the Server

Because of the way it is used, a server is classified as a service virtual machine. It is often convenient to have service virtual machines logged on automatically by the VM system as part of the IPL process. If the server is logged on automatically when the VM system is initialized, then you do not have to re-start the server when your VM system is taken down and is brought back up.

You should discuss with your VM system administrator the possibility of including your SAS/SHARE servers in the automatic log on process before you write the server's PROFILE EXEC.

If you always execute the server interactively, the PROFILE EXEC needs to set up only the virtual machine environment. However, if the server virtual machine is automatically started when your VM system is initialized, the PROFILE EXEC must set up the virtual machine environment and also must invoke the SAS System and run the PROC SERVER statement.

The following code example shows a sample PROFILE EXEC that detects whether the virtual machine is being logged on to by a person from a terminal or by the system automatically.

Example Code A2.2 Sample PROFILE EXEC

```
/*PROFILE EXEC  (written in REXX)*/
```

```
x = diagrc (24, -1)    /*Ask about virtual console*/
y = substr (x, 11, 1 /*Cond. Code 2:no real device*/

if (y = '2') then do       /*We're being AUTOLOGed*/
  'CP SPOOL CONSOLE TO * START'    /*spool console*/
  'CP SET SMSG IUCV'               /*receive SMSGs*/
  push 'SAS RUNSHARE ( LT'         /*start server */
end

else do                   /*Coming up interactively*/
  'CP TERM CHARDEL OFF' /*Turn off CHARDEL editing*/
  'CP TERM LINEDEL OFF' /*Turn off LINEDEL editing*/
end

return
```

The preceding EXEC assumes that the EXEC that is used to invoke the SAS System is named SAS. If your installation uses an EXEC with a name other than SAS to invoke the SAS System, change the preceding example, as necessary.

The sample PROFILE EXEC determines whether the virtual machine is being logged on in interactive mode or disconnected mode. It makes this determination by using the DIAGRC and SUBSTR functions, which are documented in the system product interpreter reference for the system that you are running, and DIAGNOSE code 24, which is documented in *VM/XA CP Programming Services*. In this example, the variable X is assigned information about the virtual console. The variable Y is assigned information that indicates whether the virtual console is associated with a real device. If the virtual console is not associated with a real device, the log on must result from an AUTOLOG command.

If the log on results from the AUTOLOG command, the PROFILE EXEC prepares for and begins a server session. It spools the console to the virtual machine's reader so that a record is kept of the server's activities. The EXEC then stacks a SAS command that begins a SAS session after the PROFILE EXEC finishes. The LT option is used so that the SAS log from the server session is included in the console spool file.

Placing SAS Libraries on the Server Minidisks

If the SAS data libraries that you want to share exist already, you can copy them to the server minidisks before you start the server. Use the CMS COPYFILE command or invoke SAS and perform the COPY procedure to copy the libraries. While the server is running, you can use DATA steps and procedures with the server in order to create the files that you want to share.

OpenVMS: Creating the Server Environment

You must perform the following steps to create the server environment on an OpenVMS host:

1. Set SAS performance options.
2. Create a command file for the server.
3. Run the command file for the server.
4. Run the SUBMIT command to create the server.
5. Declare the server in the DECnet database.

Setting SAS System Performance Options

The following SAS system options can be used to tune server performance:

BUFNO=*n* | *n*K | *n*M | *n*G | MAX | MIN | *hex*
specifies the number of buffers to use for SAS data sets. The default is 1.

For SAS/SHARE, setting the value of the BUFNO= option too high may hurt performance by using too much memory because SAS/SHARE may be accessing multiple files at once.

Under OpenVMS, there is no maximum number of buffers you can allocate, except for memory constraints. However, it is unusual to specify more than 10. See "All Hosts: Setting SAS System Performance Options" on page 156 for more information about the BUFNO= option.

BUFSIZE=*n* | *n*K | *n*M | *n*G | MAX | MIN | *hex*
specifies the permanent buffer size for an output SAS data set. The default is 0.

Under OpenVMS, the value of the BUFSIZE= option can range from 0 to the host maximum. The value is always rounded up to the next multiple of 512 bytes. If the value is 0, the engine picks a value based on the size of the observation.

You may want to vary the value of the BUFSIZE= option if you are trying to maximize memory usage or the number of observations per page. See "All Hosts: Setting SAS System Performance Options" on page 156 for more information about the BUFSIZE= option.

UNBUFLOG
specifies that the SAS log file (which was specified in the ALTLOG= option) is opened so that other processes can read it, and that each line written to the log is then immediately transferred to disk. This option enables you to examine the server SAS log while the server is running. The UNBUFLOG option must be specified in a SAS command or in a SAS configuration file.

Note: The overhead that is incurred by the UNBUFLOG option may degrade the performance of a busy server. △

A typical SAS configuration file follows:

```
bufno=2
bufsize=65024
unbuflog
```

See *SAS Companion for the OpenVMS Operating Environment* for more information about these SAS system options.

Creating a Command File for the Server

The command file performs any necessary process setup and invokes the SAS System. The SAS System runs a SAS program that contains any setup that is needed for the server environment and then runs the PROC SERVER statement. See Chapter 3, "Starting and Managing a SAS/SHARE Server," on page 27 for details about how to write a SAS program to start a server.

Use the following syntax to create a command file for a server:

```
$set noon
$!
$ SAS /ALTLOG=SYS$OUTPUT
      /ALTPRINT=SYS$OUTPUT
      /COMAMID=access-method
```

```
      sas-input-file
$!
$exit
```

where

ALTLOG=SYS$OUTPUT and ALTPRINT=SYS$OUTPUT
specifies the files to which SAS writes copies of the log and the procedure output, respectively. These copies of the log and the procedure output are in addition to the default .LOG and .LIS files. Specifying /ALTLOG=SYS$OUTPUT and /ALTPRINT=SYS$OUTPUT causes all SAS output from the server process to be written to the SYS$OUTPUT file, which produces a single file that contains both the OpenVMS record of the process execution and the SAS record of the server execution.

How the logical name SYS$OUTPUT is defined depends on how the command file is executed. See "Executing the Command File for the Server" on page 164 for this information.

access-method
specifies the access method that the server uses to communicate with its clients. Assign either TCP or DECNET to the COMAMID option.

sas-input-file
specifies the name of the file that contains the SAS statements to start the server. See Chapter 3, "Starting and Managing a SAS/SHARE Server," on page 27 for details about writing a program to start a server.

Executing the Command File for the Server

You can execute the command file for a server in one of two ways:

- Submit a batch job that creates a detached process by using the SUBMIT command. The detached process then executes the command file.
- If you use the DECnet access method, declare the server as a known object in the DECnet database.

Executing the SUBMIT Command to Start the Server

Use the SUBMIT command to start the server during start-up of your OpenVMS system or start a server by executing a command.

Because of its nature, a server typically runs in a detached process. Rather than execute the RUN command directly during system start-up or at other times, you should execute the RUN command in a batch command file that you submit with the SUBMIT/USER= command. This ensures that the server is created with appropriate privileges and file access authority. The SUBMIT/USER= command requires the CMKRNL privilege.

The syntax of the SUBMIT command is

$ SUBMIT/USER=*user-name batch-filename*

where

user-name
specifies the name of the user that executes the batch job that creates the process in which the server runs. The user must have the SYSNAM privilege to start the server when using the DECnet access method.

batch-filename
specifies the batch job to be executed. The purpose of the batch job is to create a detached process in which the server executes. Therefore, this batch job typically consists of one RUN command, shown as follows:

```
$ RUN /DETACHED              -
      /AUTHORIZE             -

      /INPUT=command-input-file   -
      /OUTPUT=command-output-file  -
      /ERROR=error-file   -
      /PROCESS_NAME=process-name   -
      /SYS$SYSTEM:LOGINOUT.EXE
```

where

command-input-file
specifies the name of the file that contains the commands that are executed in the detached process. For details about the contents of this file, see "Creating a Command File for the Server" on page 163.

Note: This file must also contain device or directory specifications. If the file does not contain these specifications, then the detached process may fail. △

output-file
specifies the name of the file to which the record of the execution of the detached process is written. This file should be accessible to any administrator of the server and to developers of applications that use the server. This file contains any information that is written to SYS$OUTPUT.

Note: This file must also contain device or directory specifications. If the file does not contain these specifications, then the detached process may fail. △

error-file
specifies the file to which OpenVMS errors are written. This should be accessible to any administrator of the server and to developers of applications that use the server. This file contains information that is written to SYS$ERROR.

Note: This file must also contain device or directory specifications. If the file does not contain these specifications, then the detached process may fail. △

process-name
specifies a descriptive name of the detached process in which the server executes. This value may be the same as the server name that you specify for the SERVERID= option in the PROC SERVER statement.

Declaring the Server in the DECnet Database

Use this method to start the server automatically when the first client accesses it. The advantage of this method is that you do not have to explicitly start the server. However, the disadvantage is that this method causes the first LIBNAME statement or the first PROC OPERATE statement that accesses the server to operate more slowly than it would with an explicitly started server. Notify users of the server's delayed response to these statements.

Use the Network Control Program (NCP) syntax to declare the server as a known object. For example,

```
$RUN SYS$SYSTEM:NCP
NCP> DEFINE OBJECT server-name NUMBER 0 USER user-name -
_PASSWORD user-password FILE full-filename-specification -
_PROXY NONE
```

where

server-name
specifies the name by which the server is known to the users whose SAS programs access data through it. This name must match the value that is specified for the SERVERID= option in the PROC SERVER statement and for the SERVER= option ino the LIBNAME statements in SAS programs that start and access libraries through the server.

user-name
specifies the name of the user whose User Identification Code (UIC) default directory and file protection is used to create the process in which the server executes. The user must have the SYSNAM privilege to start the server.

full-file-specification
specifies the name of the command file for the server. For a description of the contents of this file, see "Creating a Command File for the Server" on page 163. If the command file for the server is named SYS$SYSTEM:*server-name*.COM, you can omit the FILE parameter.

When you use this method to execute the command file for the server, the logical name SYS$OUTPUT is defined to the file SYS$LOGIN:NETSERVER.LOG.

It is possible to have the DECnet network automatically start the object without first issuing the NCP DEFINE command, but this method is not recommended for starting a server. Instead, you should use the NCP DEFINE command to automatically start the server. For more information about automatically starting a server without using the NCP DEFINE command, contact SAS Institute Technical Support.

OS/2: Creating the Server Environment

You must perform the following steps to create the server environment on an OS/2 host:

1. Set SAS system performance options.
2. Create a command file for the server.
3. Run the command file for the server.

Setting SAS System Performance Options

The following SAS system options can be used to tune server performance:

BUFNO=*n* | *n*K | *n*M | *n*G | MAX | MIN | *hex*X
specifies the number of buffers to use for SAS data sets. There is no maximum number of buffers you can allocate, except for memory constraints. The default is 3.

For SAS/SHARE, setting the value of the BUFNO= option too high may hurt performance by using too much memory because SAS/SHARE may be accessing multiple files at one time.

See "All Hosts: Setting SAS System Performance Options" on page 156 for more information about the BUFNO= option.

BUFSIZE=*n* | *n*K
specifies the permanent buffer size for an output SAS data set. The value can range from 512 bytes through 1 megabyte. You may want to vary the value of the BUFSIZE= option if you are trying to maximize the number of observations per page.

For SAS/SHARE, setting the value of the BUFSIZE= option too high may hurt performance by using too much memory because SAS/SHARE may be accessing multiple files at one time.

UNBUFLOG
specifies that the SAS log file (which was specified in the ALTLOG= option) is opened so that other processes can read it, and that each line written to the log is then immediately transferred to disk. This option enables you to examine the server SAS log while the server is running. The UNBUFLOG option must be specified in a SAS command or in a SAS configuration file.

Note: The overhead that is incurred by the UNBUFLOG option may degrade the performance of a busy server. △

A typical SAS configuration file follows:

```
-bufno 3
-bufsize 0
-unbuflog
```

See *SAS Companion for the OS/2 Environment* for more information about these SAS system options.

Creating a Command File for the Server

The command file performs process setup and invokes SAS, which runs a SAS program that contains the setup that is needed for the server environment and then runs the PROC SERVER statement. See Chapter 3, "Starting and Managing a SAS/SHARE Server," on page 27 for details about how to write a SAS program to start a server.

Use REXX to write a command file. The command file is appended with .CMD. A typical command file follows:

```
/* REXX exec */
string=stream('c:\server.dmp','c','query exists')
if (string='') then
   do;
     ''sas runshare $gpf dump -set sasdump
      c:\server.dmp''
   end;
else
   do;
     say ''c:\server.dmp exists -  ''
     ''you must re-name or delete it before
     starting server''
   end;
```

This command file looks for an existing dump file from a previous server execution before it invokes the SAS System. Then, the REXX EXEC invokes SAS to execute the RUNSHARE.SAS program. In addition, there may be other tasks that you need to perform before you start the server.

Running the Command File for the Server

Perform the following steps to verify that the server runs:

1 Run the server on the workstation where the SAS files to be shared are stored.

Treat the server workstation as you would a network file server machine by limiting the down time and competing work on the workstation.

2 Create an object in the start-up folder of the workstation where the server runs.

This causes the command file for the server to be executed when the OS/2 system is started. It also ensures that the server is available to the first client that attempts access. For more information about creating an object in the start-up folder, see the SAS documentation for your operating environment.

Note: Be sure that your command file does not invoke the SAS System in interactive mode. △

OS/390: Creating the Server Environment

You can invoke the SAS System from a TSO session, a batch job, or a started task. However, it is recommended that you use a started task to invoke SAS in order to run the PROC SERVER statement.

Note: If you use the cross-memory services access method, do not invoke the SAS System and create the server in a batch environment. Doing this may cause the batch initiator to drain when the server execution ends, which renders the address space (ASID) no longer usable. △

1 To start the server, create a member that is named SERVER in the system's start task procedure library that contains the JCL shown in Example Code 16.3 on page 168

Example Code A2.3 JCL in the System's Start Task Procedure Library

```
//SERVER  PROC ENTRY=entry, ID=id, SERVOPT='options', UAPW=uapw,OAPW=oapw
//SA        EXEC PGM=&ENTRY,DYNAMNBR=50mREGUIB=4096K,
//          PARM='IS=''%SHRMACS(SERVER);%STRTSRV(&ID',
//          '%STR(&SERVOPT),&UAPW,&OAPW)*,other-options')
//STEPLIB   DD DISP=SHR,DSN=&prodfix.LIBRARY
//CONFIG    DD DISP=SHR,DSN=&prodfix.CNTL(BATCHXA)
//          DD DISP=SHR,DSN=&prodfix.CNTL(SRVCNFG)
//SASAUTOS  DD DISP=SHR,DSN=&prodfix.AUTOLIB
//SASHELP   DD DISP=SHR,DSN=&prodfix.SASHELP
//SASMSG    DD DISP=SHR,DSN=&prodfix.SASMSG
//WORK      DD UNIT=SYSDA,SPACE=(6144,(500,200),,,ROUND)
//SASLOG    DD SYSOUT=*,DCB=(BLKSIZE=141,LRECL=137,RECFM=VBA)
//SASSNAP   DD SYSOUT=*
//SYSUDUMP  DD SYSOUT=*
//SYSIN     DD DUMMY
```

where

entry
is the value that is specified for the ENTRY= parameter in the cataloged procedure that is used to invoke the SAS System from a batch job. This procedure was created when base SAS software was installed. For details

about this procedure, see *Installation Instructions and System Manager's Guide for the SAS System, Version 8 under OS/390*.

id
is the server name (default or otherwise) that is passed to the PROC statement in the STRTSRV macro.

options
can be any valid option in the PROC SERVER statement. For information about PROC SERVER options, see Chapter 8, "The SERVER Procedure," on page 95.

uapw
is the user-access password for the server.

oapw
is the operator (or server administrator) password for the server.

2 Notice that the PARM= parameter uses a macro named STRTSRV to start a server. %STRTSRV is a standard SAS/SHARE autocall macro. For more information about the STRTSRV macro, see "The SAS/SHARE Macros" on page 76 and "STRTSRV" on page 141. Alternatively, you may use the SERVER macro to start a server. %SERVER executes faster than %STRTSRV. For more information about the SERVER macro, see Chapter 8, "The SERVER Procedure," on page 95.

To use %SERVER instead of %STRTSRV in the PARM= parameter, change the EXEC statement as follows:

```
//   PARM='IS=''%SERVER(&ID,&SERVOPT,&UAPW,&OAPW)'''
```

3 After the member SERVER that contains this JCL has been created, the console operator issues the following command to create the server as a started task:

```
START SERVER
```

4 When this command executes, the procedure passes the appropriate parameters to the SAS macro, STRTSRV or SERVER, which invokes the PROC SERVER statement.

5 To create a new server (one whose name is different from the ID= parameter in the JCL), enter the following:

```
START SERVER,ID=serverid
```

6 To enter the default PROC SERVER options that are indicated in the SERVOPT= parameter in the JCL, enter the following:

```
START SERVER,SERVOPT='options'
```

7 To override the user- and operator-access passwords in the START command and to override those that are specified in the UAPW= and OAPW= parameters in the JCL, enter the following:

```
START SERVER,UAPW=uapw,OAPW=oapw
```

8 To enter all of these specifications in one START command and to override those in the JCL, enter the following:

```
START SERVER,ID=serverid,SERVOPT='options'
             UAPW=uapw,OAPW=oapw
```

Invoking a SAS Program That Starts a Server

The OS/390 environment offers two methods to automatically invoke SAS and start a server:

- static program method
- macro method.

Using the Static Program Method

To use the static program method, store a SAS program that contains PROC SERVER in an external file. See Chapter 3, "Starting and Managing a SAS/SHARE Server," on page 27 for information about writing a SAS program to start a server. Invoke the SAS System by specifying the program as the primary input data stream. To use the program in the started task, locate the following line in the previous JCL code example.

```
//SYSIN DD  DUMMY
```

Change it to read as follows:

```
//SYSIN DD DSN=data-set-name,DISP=SHR
```

This method creates a server the same way each time the program runs.

Using the Macro Method

Although it is recommended that you use the STRTSRV macro from the SAS macro autocall library (see Chapter 6, "Using SAS/SHARE Macros for Server Access," on page 75), you can also create and use the SERVER macro, which executes faster than %STRTSRV.

Before you can use the SERVER macro, you must create a member named SERVER in the SAS macro autocall library and add the following statements:

```
%MACRO SERVER(id,servopt,uapw,oapw);
%*****************************************;
%* This macro invokes PROC SERVER to create *;
%* a server with the specified id.          *;
%*****************************************;
  PROC SERVER ID=&id &servopt
     %if (&uapw-=) %then
          %do;
             UAPW=&uapw
          %end;
     %if (&oapw-=) %then
          %do;
             OAPW=&oapw
          %end;
     ;
  run;
  endsas;
%MEND;
```

Setting SAS System Performance Options

These options affect the operation of the server.

Default values for these options are set in the SAS/SHARE server configuration file.

BUFNO=*n* | *n*K | *n*M | *n*G | MAX | MIN | *hex*X
specifies the number of buffers to use for SAS data sets. Under OS/390, the default value in the SAS/SHARE configuration file is 1.

For SAS/SHARE, setting the value of the BUFNO= option too high may hurt performance by using too much memory because SAS/SHARE may be accessing multiple files at one time.

The maximum number of buffers you can allocate is determined by the amount of memory available. See "All Hosts: Setting SAS System Performance Options" on page 156 for more information about the BUFNO= option.

STAE
The server should execute with the STAE option in effect so that serious but recoverable errors that are encountered by the server do not cause it to terminate abnormally.

SVC0SVC=
SVC0R15=
The SAS SVC Routine 0 is required for SAS/SHARE software. You must specify the SAS system options SVC0SVC= and SVC0R15= to accurately reflect the way the SVC was installed. Get this information from the person who installed base SAS software.

MEMLEAVE
New for Version 7 and Version 8, this option stores supplemental memory storage, which is a dynamic amount of space in a user's available region. The MEMSIZE= option is reserved for cleanup that is required when SAS has used all available memory and abends. Setting MEMLEAVE is useful at sites that run memory-intensive applications.

If MEMSIZE= is not set explicitly in a configuration file or at SAS invocation, MEMLEAVE is set automatically to a value that is equal to the user's available region.

MEMSIZE=*n* | *n*K | *n*M
For Version 6, specifies a limit on the total amount of memory that SAS uses at any one time. The server should be allowed a large amount of memory in order to accommodate the needs of all its users. The number of concurrent SAS files open by server users and the number and size of I/O buffers for each SAS file have a direct impact on the amount of memory that is required by the server. You should be aware that, in some cases, SAS terminates abnormally when it is unable to satisfy a memory request. The default is 0.

A change from Version 6 behavior, the desired effect of the MEMSIZE= option for Version 7 and Version 8 is to allocate extended storage. Extended storage is a small amount of a user's address space that is reserved for cleanup when SAS has used all available memory and abends. Rather than to assign an explicit value to MEMSIZE=, SAS automatically assigns to it the current value of MEMLEAVE.

You may continue to use the Version 6 implementation of MEMSIZE= and assign a value to it. Alternatively, you may use the Version 7 and Version 8 implementation instead.

For the Version 7 and Version 8 implementation, you can set the MEMSIZE= option at SAS invocation only. Do not set MEMSIZE= in a configuration file, in an OPTIONS statement, or in an AUTOEXEC file.

For the Version 6 implementation, you can use the following values with the MEMSIZE= option:

n
specifies a number from 0 to 2,147,483,648. A value of 0 indicates that there is no limit except the operating environment limit, which may be site-specific.

*n*K
specifies the number of kilobytes, from 0 to 2,097,152.

*n*M
specifies the number of megabytes, from 0 to 2048.

In order to limit the amount of virtual memory that SAS can use, you should specify a value for the MEMSIZE= option. For SAS/SHARE, you may want to specify a value of 36M for the MEMSIZE= option to allow for peak usage periods.

VMCTLISA=*value*
specifies the size of the ISA (initial storage allocation) for SAS memory management and control blocks. This memory is critical to SAS software execution.

You should specify a VMCTLISA= value of 1536K to ensure that SAS memory management routines have sufficient memory for their control blocks.

VMPAISA=*value*
VMPAOSA=*value*
VMTAISA=*value*
VMTAOSA=*value*
specifies the size of the memory above the 16-Mb line.

VMPAISA specifies the size of the ISA for permanent memory above the 16-Mb line. VMPAOSA specifies the size of the OSA (overflow storage allocation) for permanent memory above the 16-Mb line.

VMTAISA specifies the size of the ISA for temporary memory above the 16-Mb line. VMTAOSA specifies the size of the OSA for temporary memory above the 16-Mb line.

The bulk of server memory is allocated above (A) the 16-Mb line. The initial storage allocation should be sizeable. Specify VMPAISA=7680K and VMTAISA=7680K to enable SAS software to satisfy many memory requests without incurring too much memory fragmentation. Secondary allocations (OSA) should be large enough to hold down the number of times that SAS software has to request more memory from the operating system but not so large that the requests fail due to memory fragmentation. Specify VMPAOSA=3072K and VMTAOSA=3072K.

VMPBISA=*value*
VMPBOSA=*value*
VMTBISA=*value*
VMTBOSA=*value*
specifies the size of memory below the 16-Mb line.

VMPBISA specifies the size of the ISA for permanent memory below the 16-Mb line. VMPBOSA specifies the size of the OSA for permanent memory below the 16-Mb line. VMTBISA specifies the size of the ISA for temporary memory below the 16-Mb line. VMTBOSA specifies the size of the OSA for temporary memory below the 16-Mb line.

The server uses relatively little memory below (B) the 16-Mb line. Specify VMPBISA=768K, VMPBOSA=384K, VMTBISA=96K, and VMTBOSA=96K.

These options can be specified in a SAS configuration file. A typical SAS configuration file entry follows:

```
vmctlisa=1536K
vmpaisa=7680K
vmpaosa=3072K
vmtaisa=7680K
vmtaosa=3072K
```

```
vmpbisa=768K
vmpbosa=384K
vmtbisa=96K
vmtbosa=96K
procleave=150K
sysleave=150K
```

The omission of the MEMSIZE= option from the configuration file implies that SAS sets the MEMLEAVE option to a value that is equal to the amount of a user's available region.

See *SAS Companion for the OS/390 Environment* for more information about these SAS system options.

UNIX: Creating the Server Environment

You can invoke the SAS System and start a server session automatically by using the following command syntax:

```
sas sas-input-file server-config-file
```

where

sas-input-file
is the name of the file that contains the SAS statements to start a server. For information about the content of this file, see Chapter 3, "Starting and Managing a SAS/SHARE Server," on page 27.

server-config-file
is the file that contains the system option settings for the server's SAS session. For information about the SAS system options, see the next section.

Include the SAS command in a start-up script so that the server is created at machine start-up. Doing so ensures that the server is running whenever a client needs it.

Setting SAS System Performance Options

These options affect the operation of the server.

BUFNO=*n* | *n*K | *n*M | *n*G | MAX | MIN | *hex*X
specifies the number of buffers to use for SAS data sets. The default is 1.

For SAS/SHARE, setting the value of the BUFNO= option too high may hurt performance by using too much memory because SAS/SHARE may be accessing multiple files at once.

The maximum number of buffers you can allocate is determined by the amount of memory available. See "All Hosts: Setting SAS System Performance Options" on page 156 for more information about the BUFNO= option.

BUFSIZE=*n* | *n*K | *n*M | *n*G | MAX | MIN | *hex*X
specifies the permanent buffer size for an output SAS data set. Under UNIX, the default is 0.

For SAS/SHARE, setting the value of the BUFSIZE= option too high may hurt performance by using too much memory because SAS/SHARE may be accessing multiple files at once.

If you use the default value 0 when you create a SAS data set, the V7 engine calculates a buffer size to optimize CPU and I/O use. This size is the smallest multiple of 8K that can hold 80 observations but is not larger than 64K.

See "All Hosts: Setting SAS System Performance Options" on page 156 for more information about the BUFSIZE= option.

FILELOCKS NONE | FAIL | CONTINUE
specifies whether SAS locks others out of a SAS file before opening it for update or output. This prevents other SAS sessions from opening it for update or output at the same time. Because a server's SAS session is no different from any other SAS session, it is recommended that you specify FILELOCKS FAIL in all SAS sessions to prevent a server and any other SAS session from simultaneously opening the same SAS file for update or output. The default value is FAIL.

The values for FILELOCKS are

NONE
specifies that no file locking occurs. All protection from shared update access to SAS files is removed. This is the default.

FAIL
specifies that file locking is in effect. If a file cannot be locked, an attempt to open it fails.

CONTINUE
specifies that file locking is in effect. If a file is already locked by someone else, an attempt to open it fails. If the file cannot be locked for some other reason, the file is opened and a warning message is sent to the log.

For file locking to work on some hosts, you must have the host's file locking service running. This usually involves having a lock daemon (such as **lockd**) and a stat daemon (such as **statd**) running. You may also need to execute other commands. For details, see the manual pages or system administration instructions.

In addition, if you are working with NFS-mounted files, the file locking service must be running both on your client and on the server.

MEMSIZE=*n* | *n*K | *n*M | *n*G | MAX | MIN | *hex*X
specifies a limit on the total amount of memory that SAS uses at any one time. Under UNIX, the default is 32M.

The server should be allowed a large amount of memory in order to accommodate the needs of all its users. The number of concurrent SAS files open by server users and the number and size of I/O buffers for each SAS file have a direct impact on the amount of memory required by the server. You should be aware that, in some cases, SAS terminates abnormally when it is unable to satisfy a memory request.

The MEMSIZE= option takes the following values:

n
specifies the amount of memory in bytes.

*n*K
specifies the amount of memory in 1-kilobyte multiples.

*n*M
specifies the amount of memory in 1-megabyte multiples.

*n*G
specifies the amount of memory in 1-gigabyte multiples.

MIN
specifies 0, which means that there is no limit.

MAX
specifies 2,147,483,647. On 64-bit systems, MAX is 9,007,199,254,740,992.

Note: The default may be larger on some machines. △

*hex*X
specifies the amount of memory in a hexadecimal number that must be followed by an X.

UNBUFLOG
specifies that the SAS log file (which was specified in the ALTLOG= option) is opened so that other processes can read it, and that each line written to the log is then immediately transferred to disk. This option enables you to examine the server SAS log while the server is running. The UNBUFLOG option must be specified in a SAS command or in a SAS configuration file.

Note: The overhead that is incurred by the UNBUFLOG option may degrade the performance of a busy server. △

A typical SAS configuration file follows:

```
-bufno 1
-bufsize 0
-filelocks fail
-memsize 32M
-unbuflog
```

See *SAS Companion for UNIX Environments* for more information about these SAS system options.

Windows: Creating the Server Environment

You can invoke the SAS System and start a server session automatically by adding a program item to your Windows StartUp Program Group as follows:

1. Highlight the StartUp Program Group in the Windows Program Manager.
2. Select **New ...** from the **File** menu.
3. Select **Program Item** from the New Program Object dialog box.
4. Click on **OK**.
5. Complete the Program Item Properties dialog box to reflect your site's configuration. The following sample properties assume that SAS is located in the C:\SAS directory and the program that is used to invoke the server is named **server.sas**. Sample properties follow:

   Description:
   Server Startup

   Command Line:
   c:\sas\sas c:\sas\server $logflush $gpf dump -set sasdump

   Working Directory:
   c:\sas

Shortcut Key:
None

6 You can use the Command Line field to specify SAS system options. For example, the following command line invokes SAS, starts a server, and specifies that a memory dump can be written to C:\SAS\SERVER.DUMP whenever a general protection fault occurs:

```
c:\sas\sas c:\sas\server $logflush $gpf dump
   -set sasdump c:\sas\server.dmp
```

7 Click on **OK**.

These steps add an application icon for the SAS System to the StartUp Program Group and start the server session when you re-start Windows.

Setting SAS System Performance Options

The following SAS system options can be used to tune server performance:

BUFNO=*n* | *n*K | *n*M | *n*G | MAX | MIN | *hex*
specifies the number of buffers to use for SAS data sets. There is no maximum number of buffers you can allocate, except for memory constraints. Under Windows, the default is 3.

For SAS/SHARE, setting the value of BUFNO= option too high may hurt performance by using too much memory because SAS/SHARE may be accessing multiple files at one time.

See "All Hosts: Setting SAS System Performance Options" on page 156 for more information about the BUFNO= option.

BUFSIZE=*n* | *n*K
specifies the permanent buffer size for an output SAS data set. The value can range from 512 bytes through 1 megabyte. The value can be specified as *n* buffers or *n*K (kilobyte) buffers, where K=1,024. You may want to vary the value of the BUFSIZE= option if you are trying to maximize the number of observations per page. Under Windows, the default is 0.

For SAS/SHARE, setting the value of the BUFSIZE= option too high may hurt performance by using too much memory because SAS/SHARE may be accessing multiple files at one time.

UNBUFLOG
specifies that the SAS log file (which was specified in the ALTLOG= option) is opened so that other processes can read it, and that each line written to the log is then immediately transferred to disk. This option enables you to examine the server SAS log while the server is running. The UNBUFLOG option must be specified in a SAS command or in a SAS configuration file.

Note: The overhead that is incurred by the UNBUFLOG option may degrade the performance of a busy server. △

A typical SAS configuration file follows:

```
-bufno 3
-bufsize 0
-unbuflog
```

See *SAS Companion for the Microsoft Windows Environment* for more information about these SAS system options.

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