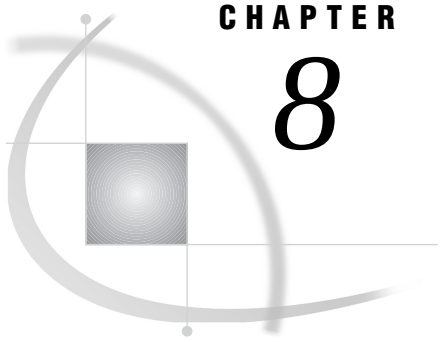

CHAPTER

8

The SERVER Procedure

Introduction

You invoke the SERVER procedure to start a SAS/SHARE server, which enables multiple users to access and use SAS data libraries and members in those libraries at the same time. As part of server start-up, you may assign a server name, and you may specify optional properties such as permission for clients to define SAS data libraries to a server, a report of clients' resource consumption, and a password that a user must supply to access the server.

You can invoke the SERVER procedure using any of the SAS methods of processing: noninteractive mode, interactive line mode, display manager mode, or batch mode. Production SAS/SHARE servers are typically run in batch mode. For interactive testing, PROC SERVER is typically run in interactive line mode.

Note: If you use display manager, which is not recommended for running PROC SERVER, enter the display manager command AUTOSCROLL 1 on the command line of the Log window. △

For details about how to invoke the SERVER procedure and for an example of a SAS log, see Chapter 3, "Starting and Managing a SAS/SHARE Server," on page 27.

Syntax

PROC SERVER <*options*>;
ALLOCATE SASFILE *SAS-data-set1* (*data-set-options*)
<*SAS-data-set2* (*data-set-options*) ...
SAS-data-set20 (*data-set-options*) >;

Syntax Descriptions

The PROC SERVER statement options are

ALLOC | NOALLOC
determines whether clients can define additional SAS data libraries to a server after a server has started. The ALLOC option allows clients to define libraries to a server. The NOALLOC option prevents clients from defining additional libraries to the server and restricts clients to accessing libraries that are defined by a server administrator. The default is ALLOC.

ACCTLVL=*value* | (*value 1* < ... *value n* >)
allows you to specify when a server prints accounting data. This option is useful for "fine-tuning" an application, because it allows you to examine in more detail how different parts of the application use server resources. The default is USER.

The ACCTLVL= and LOG= options are related in that the ACCTLVL= option specifies when accounting information is logged, and the LOG= option specifies which accounting information is logged.

Value for ACCTLVL= may be

ALL
Accounting information is written to the server's SAS log file at all of the specified times (DATA, USER, and RESOURCE_ENVIRONMENT).

DATA
Accounting information is written to the server's log when a SAS file is closed by a user. This data shows the usage of server resources that resulted from the use of that file by that user.

USER
Accounting information is written to the server's log when a user disconnects from the server. This data shows the usage of server resources that resulted from all of that user's activity while connected to the server. This is the default.

RESOURCE_ENVIRONMENT
Accounting information is written to the server's log when a "resource environment" is terminated by a user. Examples of resource environments include a SAS procedure, a SAS window, a DATA step, a SAS process, or other internal SAS activity. This data shows the usage of server resources that resulted from requests made in the context of that resource environment by that user.

AUTHENTICATE=REQUIRED | REQ| OPTIONAL | OPT
The AUTHENTICATE= option controls whether a server requires connecting users to supply a valid userid and password when they connect to the server. The default is REQUIRED.

This option is used with a communications access method security option (for example, use the TCPSEC= option for TCP/IP). When the value of the AUTHENTICATE= option is REQUIRED, the security options of all the access methods that are used by the server must be set to _SECURE_. Also, PROC SERVER will not start up unless all of the access methods guarantee that they require a valid userid and password combination before establishing a connection between a client and the server.

When the value of the AUTHENTICATE= option is OPTIONAL, the communications access method security options may or may not be set to _SECURE_. Each access method may have its security option set differently. This

situation can be useful for a site that trusts the identities of users that connect through one access method but does not trust the identities of users that connect through another access method. In this case, only the access method by which the non-trusted clients connect may have its security option set to _SECURE_, and the value of the AUTHENTICATE= option is set to OPTIONAL to allow the trusted users to connect without requiring authentication.

When an access method supplies an authenticated userid (typically, by requiring a connecting client to specify a valid userid and password combination), the server uses that validated identity to verify the user's authorization to access SAS files. When the value of the AUTHENTICATE= option is OPTIONAL and some access methods do not supply validated userids, those clients are allowed to access all SAS files that the server is allowed to access. In this situation, you should run the server under a userid that does not have open access to the files on its computer.

DTF=*SAS-datetime-format* |_NODTS_
DTFORMAT=*SAS-datetime-format*|_NODTS_
specifies the format for the date-time stamp at the beginning of each message that is written to the server's log. You can specify any SAS date, time, or date and time format; or you can specify your own date and time format. The default is DATETIME22.3. For details about specifying these formats, see the chapter about SAS formats in *SAS Language Reference: Dictionary*.

If you specify your own format for this option, the number to format that is supplied to the formatting routine is a SAS datetime value. For an example of the date and time format in the server's log, see "Server Log Reporting All Logging Statistics" on page 39.

Specifying the value _NODTS_ suppresses the date-time stamp.

LRPYIELD=*value*
indicates to a long-running process in the server's SAS session how frequently the process should yield so that others can use the server. The default is 10000; this value has no units. Increase the value to have a long-running process yield more frequently; decrease the value to have it yield less frequently. Setting the value to 0 does not yield at all. The server can process other requests only after the long-running process has finished.

An example of a long-running process is when a client accesses a PROC SQL view that joins two large data sets that require the server to sort the data sets. Another example is when a client issues a WHERE clause that causes the server to search millions of observations sequentially to find the first one that satisfies the WHERE clause.

LOG=*value*|(*value 1 <. . . value n>*)
specifies that a server generate statistics about client-server transactions. Therefore, clients can be charged for their share of the server's operation. If specified, this option causes the server to write a line to its log for each resource that is consumed when a client disconnects from the server.

You can specify multiple values for the LOG= option. To do this, enclose the values in parentheses and separate them with either a space or a comma; for example, LOG=(MESSAGE BYTECOUNT).

Note: The values IO and CPU are no longer accepted. The data that was reported by the options in Version 6 offered limited accuracy, and the changes in resource tracking in Version 7and Version 8 reduce the potential accuracy even further. △

Note: If you previously set the SAS options STIMER or FULLSTIMER because you had specified a LOG= option value of IO or CPU, you can re-set or delete these SAS options in order to improve your server's performance. △

Value for LOG= may be

ALL
reports the types of server resources that are described by the following option values: ACTIVETIME, BYTECOUNT, ELAPSEDTIME, MESSAGE, and QUERY.

ACTIVETIME
ACTIVE
directs the server to write to its log the cumulative elapsed time that the server has processed requests on behalf of a client in a single session. This value is printed by using the SAS format TIME15.3.

BYTECOUNT
BYTE
directs the server to write to its log the number of cumulative bytes in client messages to a server in a single session.

The value is automatically re-scaled prior to printing, and the appropriate character - K (when the count reaches ten million bytes), M (when the count reaches ten million K bytes), and G (when the count reaches ten million M bytes) - is appended to the value, which is printed with the SAS format COMMA10.0.

ELAPSEDTIME
ELAPSED
directs the server to write to its log the amount of time that elapsed during the recorded event (for example, the length of time a file was open or the length of time that the user was connected to the server). This value is printed by using the SAS format TIME15.3.

MESSAGE
MSG
directs the server to print in its log the number of messages that are exchanged with each client. A *message* is a request from a client session and a corresponding response from the server. The server prints the number of messages that are received and the number of messages that are rejected. A server rejects messages because of stopped or quiesced resources or because of errors in the software, such as incorrect requests from a client. This value is printed with the SAS format COMMA15.0. For an example of the MESSAGE option in the server log, see Output 3.2 on page 39.

Note: Do not confuse the MESSAGE value for the LOG= option with the option MSGNUMBER. △

QUERY
directs the server to write to its SAS log each SQL query that it receives through Remote SQL Pass-Through from a SAS session or other client. By default, the server logs update and output SQL statements but not queries.

An example follows:

```
22JUL97:13:57:03 JSMITH(27) in
''SQL(0) has issued select region,year,
ytd from yearly.regsales where
country='USA' to SQLVIEW.
```

For a typical server log, see Output 3.2 on page 39.

NORMTVIEW
disables the ability of a server to interpret SAS data views.

By default, a SAS data view is interpreted in the server SAS session, and the data that is produced by the view is transmitted to a client SAS session.

Occasionally, it is preferable to transmit the view - the instructions for producing the data - to a client SAS session and to have the view interpreted and the data assembled in the client SAS session. The RMTVIEW= option of the LIBNAME statement can be used to request this on a library-by-library basis.

The NORMTVIEW option enforces this action (transmission of the view instead of the data) for all users of the server, regardless of whether each user specifies the RMTVIEW= option in a LIBNAME statement.

This option is made available for specific needs and is only rarely appropriate. If you think that this option may be useful for a server, contact SAS Institute Technical Support to review the circumstances.

SERVER=*serverid*
SERVERID=*serverid*
ID=*serverid*
specifies a name for the server. The server name must meet the criteria for a valid SAS name, although it can also include the following characters: dollar sign ($), at sign (@), and pound sign (#). The default for this option is host-dependent; it is the name of the host from which SAS was invoked. However, you may assign any valid SAS name to the server.

specifies a name for the server. The server name must meet the criteria for a valid SAS name, although it can also include the following special characters: dollar sign ($), at sign (@), and pound sign (#).

Server naming is also constrained by the host type and the access method that you specify for communication between a server and a client session. For complete information about how to name servers by host, see *Communications Access Methods for SAS/CONNECT and SAS/SHARE Software.*

MSGNUMBER
MSGN
associates a unique message number (represented in hexademical notation) with each type of operation for which a message is recorded in a server log. A typical message written to a server log follows:

```
06JAN1999:08:28:20.911 043131 SAS server SHR1 started
```

The message number **043131** identifies a PROC SERVER start-up operation.

Message numbers are useful for server log analysis applications. Such a program could count the number of instances of an operation that occur in a given client-server session. Collection and analysis of such statistics may help you with server load balancing. SAS provides a set of server log analysis program prototypes that you can customize for your needs. Among these prototypes is a file that maps message numbers to operations. See “Introduction” on page 87 for information about the server log analysis tools.

Note: Because message numbers can change from one software release to another, avoid hardcoding message numbers in your applications. You may use macros instead. △

Note: Do not confuse the option MSGNUMBER with the MESSAGE value for the LOG= option. △

OAPW=*password*
specifies a password that the server administrator must supply (by using the OPERATE procedure) to connect to the server. The password must be a valid SAS name. The value for this option is replaced by Xs in the server log. To protect this password, you should use your site's security software to limit access to the SAS program statements that are used to create the server.

PT2DBPW=*password*
specifies the password that is required of a client that runs the SQL CONNECT TO REMOTE statement with the DBMS= option. It is used to restrict the client's ability to connect through a SAS/SHARE server to another data server (including a DBMS or another SAS/SHARE server) to only those users who know the password.

For details about how the client supplies this password to an SQL CONNECT TO REMOTE statement, see Chapter 10, "Remote SQL Pass-Through (RSPT) Facility," on page 109.

TBUFNO=*value*
specifies the number of transmission buffers that are used by a server to receive data from and to reply with data to users. The default is 4.

A transmission buffer is assigned while a server is processing a request from a user. If a server is asked by users to perform many long-running requests, transmission buffers can be tied up, making it necessary to increase the value of this option. In most cases, the value of this option can remain much smaller than the number of users that are simultaneously connected to the server.

There is an interaction between this option and the LRPYIELD= option. If the value of the LRPYIELD= option is high, long-running requests from users tend to yield more frequently, which provides more opportunity for new requests from users to be received by the server. This can result in transmission buffers being tied up by long-running requests that are yielding frequently (and therefore taking even longer to run). This situation may suggest increasing the value of the TBUFNO= option.

The amount of memory that is used by a server for transmission buffers can be calculated by multiplying the values of the TBUFNO= and TBUFSIZE= options.

TBUFSIZE=*value*
specifies the default size of a buffer that the server uses for transmitting multiple observations. The server uses this value when computing the number of observations to transmit in each multi-observation transfer between the server and the client sessions. The default is 32K.

You cannot calculate the number of observations per transfer by dividing the observation length into the value that you specify for the TBUFSIZE= option. To determine the effect of this option on your data sets, use the PROC SERVER option LOG=MESSAGE and compare the number of messages exchanged between the server and the client sessions as a function of the value of the TBUFSIZE= option and the data set's observation length.

Transfers of multiple observations are used only for data sets that are opened for output or for certain types of input. Multi-observation transfers are not used for data sets that are opened for update.

If the minimum amount of data that must be transferred exceeds the specified buffer size, the buffer size increases to allow the transfer. Specifying an excessive value for this option may cause your server or clients to run out of memory and, in turn, to terminate abnormally.

UAPW=*password*
specifies a password that a client must supply in the LIBNAME statement to establish communication with the server. The password must be a valid SAS name. The value for this option is replaced by Xs in the server log. To protect this

password, you should use your site's security software to limit access to the SAS program statements that are used to create the server.

Examples

The following are examples of how to start a server:

```
proc server id=share1;
proc server id=share1 noalloc;
proc server id=share1 log=msg;
```

The first statement starts the server SHARE1. The second statement starts the server SHARE1 that does not allow clients to define SAS data libraries to the server. The third statement starts the server SHARE1 and reports all message counts to the server log.

ALLOCATE SASFILE Command

Note: A statement that is used in the SERVER procedure is called a command. △

The SERVER procedure is interactive; that is, its statements are executed as they are encountered. The SERVER procedure executes until it is terminated by a QUIT or a RUN command.

You can use the ALLOCATE SASFILE command to specify that one or more SAS data sets remain open for the duration of a server session until you terminate the server.

Keeping SAS data sets open until you terminate the server can enhance server performance by reducing overhead as users open and close the SAS data sets during the server's processing. Files kept open this way are available for users to read and update, but they are not available for exclusive access. Actions that require exclusive access to a SAS data set include using a DATA step to replace it and using PROC DATASETS to rename variables in it.

Syntax

ALLOCATE SASFILE *SAS-data-set1* (*data-set-options*)
< *SAS-data-set2* (*data-set-options*) ...
SAS-data-set20 (*data-set-options*)>;

SAS-data-set
descriptor information and its related data values organized as a table of observations and variables that can be processed by SAS.

data-set-options
specify actions that apply only to the SAS data set with which they appear. For complete details about data set options, see *SAS Language Reference: Dictionary*.

You can specify up to 20 SAS data set names in each ALLOCATE SASFILE command.

You must use either a LIBNAME statement or an external method to define all librefs before you execute the ALLOCATE SASFILE command. SAS data sets that are specified in an ALLOCATE SASFILE command must exist before the ALLOCATE SASFILE command is executed. SAS data sets that are specified in an ALLOCATE SASFILE command may be read or updated by users of the server. Other kinds of SAS

files (for example, CATALOG) may not be specified in an ALLOCATE SASFILE command.

Users can access SAS data files through the server regardless of whether the data files are specified in ALLOCATE SASFILE commands. When several users access the same file through a server, the overhead of opening the file for the first access is greater than the overhead for the additional accesses. The ALLOCATE SASFILE command provides a way to ensure that the overhead of opening specific SAS files the first time is incurred only one time (when the server is starting up). While the server is running, less work is required of the server each time a client opens the file.

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