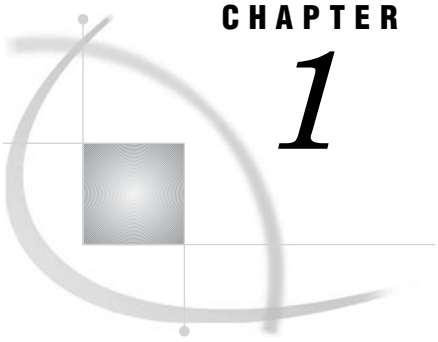

CHAPTER

1

Initializing and Configuring SAS Software

Invoking SAS in the z/OS Environment

Invocation Methods

You can invoke SAS with any of the following methods:

- in interactive mode under TSO using the SAS CLIST
- in interactive mode under TSO using the SASRX exec
- in batch mode with the SAS cataloged procedure
- by logging on to SAS directly and bypassing the TSO terminal monitor program.

Invoking SAS under TSO: the SAS CLIST

To invoke SAS under TSO, you execute the SAS CLIST by typing a command (usually **SAS**) at the READY prompt. The SAS CLIST is an external file that contains TSO commands and control instructions.

At each site, the name of the command that you use and the SAS CLIST might have been modified by your on-site support personnel. Ask your support personal for site-specific information about the CLIST.

The SAS CLIST starts a SAS windowing environment session, an Explorer session, an interactive line mode session, or a noninteractive session, depending on the defaults that have been specified in the CLIST. To override the mode of running SAS that is specified in the CLIST, you use commands similar to those shown in Table 1.1 on page 6. (Again, the exact commands that you use might be site-specific.)

Invoking SAS under TSO: the SASRX exec

SASRX is a REXX program that you can use to invoke SAS. It is provided as an alternative to the SAS CLIST. SASRX supports the same command-line syntax as the SAS CLIST. It also supports these additional features:

- mixed-case option values
- additional options
- UNIX-style option specification
- direct specification of SAS system options
- UFS file and directory names as option values

At each site, the command that you use and the SASRX exec itself might have been modified by your on-site SAS support personnel. Ask your support personnel for

site-specific information about the SASRX exec. For details about the SASRX exec, see Appendix 1, "Starting SAS with SASRX," on page 641.

Throughout this document, references to the SAS CLIST apply equally to the SASRX exec.

Commands for Invoking SAS

The following table contains examples of commands that you can use to invoke SAS.

Table 1.1 Commands for Invoking SAS

Mode	To Invoke	To Terminate	Description
SAS windowing environment	**sas options('dms')** or **sasrx -dms**	**bye** or **endsas**	enables you to write and execute SAS programs and to view the SAS log and SAS procedure output in an interactive windowing environment. If this is the default at your site, then you can invoke it by entering **sas** (or **sasrx**) with no options.
Explorer	**sas options('explorer')** or **sasrx -explorer**	**bye** or **endsas**	enables you to manipulate SAS data and files visually, launch SAS applications, and access SAS windowing environment windows and Output Delivery System hierarchies.
interactive line mode	**sas options('nodms')** or **sasrx -nodms**	**/*** or **endsas;** statement	prompts you to enter SAS statements at your terminal, one line at a time.
noninteractive mode	**sas input(''''my.sas.program'''')** or **sasrx -input 'my.sas.program'**	not applicable	executes interactively, but it is called noninteractive because the program runs with no intervention from the terminal.

Invoking SAS in Batch Mode: the SAS Cataloged Procedure

To invoke SAS during a batch job, use a JCL EXEC statement that executes the SAS cataloged procedure that invokes SAS, such as

```
//MYJOB EXEC SAS
```

By specifying parameters in the JCL EXEC statement, you can modify the way in which SAS is invoked.

At each site, the JCL EXEC statement that you use and the cataloged procedure itself might have been modified by your on-site SAS support personnel. Ask your support personnel for site-specific information.

Logging On to SAS Software Directly

z/OS sites can choose to substitute SAS for the standard TSO terminal monitor program, enabling users to log on to SAS directly. If SAS comes up automatically when you log in, then your system might have already been set up to log on to SAS directly.

By automatically invoking SAS software or a SAS application when users log on, site administrators can insulate users from the TSO environment. Because SAS is running as its own terminal monitor program, TSO commands are not accessible to users. This reduces memory usage slightly.

This method of invoking SAS also provides the following advantages:

- Sites can restrict user access to the TSO environment.
- Novice users do not have to learn how to work in the TSO environment.

Your on-site SAS support personnel can find complete information about this method of invoking SAS in the installation instructions for SAS in the z/OS environment.

What If SAS Doesn't Start?

If SAS does not start, the SAS log might contain error messages that explain the failure. Any error messages that SAS issues before the SAS log is initialized are written to the SAS Console Log, which is the SASCLOG ddname destination. Under TSO, the SASCLOG ddname destination is normally the terminal, but the destination might be changed by the on-site SAS support personnel by changing the CLIST or SASRX exec that invoked SAS. Similarly, a batch job or started task normally assigns the SASCLOG ddname to a spooled SYSOUT class, but the destination might have been changed by the on-site SAS support personnel by changing the catalog procedure used to invoke SAS. Spooled SYSOUT data can be printed or viewed online with a JES spool viewer such as SDSF, IOF, or EJES.

Connecting to SAS under z/OS

Under z/OS, you can access or connect to a SAS session in any of the following ways:

3270 terminals
: You can use devices that support extended data streams as well as those that do not. See "Terminal Support in the z/OS Environment" on page 209 for more information about terminal support.

terminal emulators
: Terminal emulators that you can use to access SAS on z/OS include Attachmate Extra!, Hummingbird Host Explorer, and others.

  Note: SAS best supports those terminal emulators that closely conform to the original IBM specifications for the 3270 terminal. If you are having difficulties with the SAS vector graphics in your emulator session, make sure that the settings for your emulator match these specifications as closely as possible. △

SAS/CONNECT software
: SAS/CONNECT supports cooperative and distributed processing between z/OS and Windows, and UNIX, and OpenVMS. It supports the TCP/IP (Transmission Control Protocol/Internet Protocol) and XMS (Cross-Memory Services) communications access methods, which enable local clients who are running SAS to communicate with one or more SAS applications or programs that are running in remote environments. For more information, see *Communications Access Methods for SAS/CONNECT and SAS/SHARE*.

SAS/SHARE software
: SAS/SHARE enables local and remote clients in a heterogeneous network to update SAS data concurrently. It also provides a low-overhead method for multiple

remote clients to read local SAS data. For more information, see *SAS/SHARE User's Guide*.

SAS/SESSION software
SAS/SESSION enables terminal users who are connected to the Customer Information Control System (CICS) to communicate with SAS software in a z/OS environment. It uses the LU6.2 (APPC/MVS) protocol. Your on-site SAS support personnel can find more information about SAS/SESSION in the installation instructions for SAS software in the z/OS environment.

SAS Intelligence Platform
SAS creates and delivers enterprise intelligence through the SAS Intelligence Platform. This cohesive platform is based on an architecture that fully integrates SAS technologies in data extraction, transformation, and loading; data storage; business intelligence; and analytics. These capabilities provide the end-to-end infrastructure necessary for exploring, analyzing, optimizing, reporting, and understanding your data. For more information, see the documentation about "SAS Intelligence Platform: Administration Documentation" at **support.sas.com**.

Customizing Your SAS Session

Overview of Customization

Whether you are using interactive processing under TSO or batch processing, you might want to customize certain aspects of your SAS session. For example, you might want to change the line size or page size for your output, or you might want to see performance statistics for your SAS programs.

You can customize your SAS sessions by setting SAS system options that control SAS behavior. For more information about SAS system options, see Chapter 18, "System Options under z/OS," on page 471.

Customizing Your SAS Session at Startup

You can customize your SAS session in five ways:

- Under TSO, pass operands into the SAS CLIST or SASRX exec that your site uses to invoke SAS. (See "Invoking SAS under TSO: the SAS CLIST" on page 5.) This method is usually used for one-time overrides of CLIST or SASRX exec operands. Here are three examples:
  - `sas options('nocenter linesize=80')`
  - `sasrx -options (nocenter linesize=80)`
  - `sasrx -nocenter -linesize 80`
- In batch mode, pass parameters into the SAS cataloged procedure that your site uses to invoke SAS. (See "Invoking SAS in Batch Mode: the SAS Cataloged Procedure" on page 6.) This method is usually used for one-time overrides of parameters in the cataloged procedure. Here is an example:

```
//MYJOB  EXEC SAS,
//   OPTIONS='NOCENTER, LINESIZE=80'
```

- Specify SAS system options in a user configuration file. (See "Configuration Files" on page 9.) This method is useful if you, as an individual user, always want to

override the values of system options that are specified in your site's system configuration file. The following examples use a TSO command to specify a user configuration file:

```
sas  config(''my.config.file'')
```

or

```
sasrx  -config 'my.config.file'
```

The following example specifies a user configuration file with JCL:

```
//MYJOB  EXEC SAS,
//       CONFIG='MY.CONFIG.FILE'
```

- Execute SAS statements (such as OPTIONS, LIBNAME, and FILENAME statements) in an AUTOEXEC file. (See "Autoexec Files" on page 11 for more information.) This method is most useful for specifying options and allocating files that pertain to a particular SAS application.
- In interactive mode, specify a SASUSER library that contains a user Profile catalog. (See "SASUSER Library" on page 12.)

See "Precedence for Option Specifications" on page 17 for information about the order of precedence for options specified using these methods.

Configuration Files

Overview of Configuration Files

A *configuration file* contains SAS system options that are set automatically when you invoke SAS. SAS uses two types of configuration files:

- the system configuration file, which is used by all users at your site by default. Your on-site SAS support personnel maintain the system configuration file for your site.
- a user configuration file, which is generally used by an individual user or department.

Creating a User Configuration File

To create a user configuration file, use any text editor to write SAS system options into a physical file. The configuration file can be either a sequential data set or a member of a partitioned data set. It can have any record length, and either fixed length or variable length records. Embedded configuration files (those specified via the CONFIG= option inside a configuration file) can also be in a UFS directory.

Format of a Configuration File's Contents

Each line of a configuration file can contain one or more system options or comments. If you specify more than one system option on a line, use either a blank space or a comma to separate the options. If the file has the legacy configuration file format of LRECL=80 and RECFM=FB, then only columns 1-72 are used. The contents of columns 73-80 are ignored. If the file has any other format, then the entire line is used.

Two different types of comments are supported. If a line contains an asterisk in column one, then the entire line is a comment. If a comment of this type requires multiple lines, each line must begin with an asterisk. Comments beginning with /* and

ending with */ can appear anywhere between option specifications, but cannot be imbedded within an option specification. Comments of this type can continue across line boundaries.

Note: An */ that ends a comment cannot be in column one. If it is in column one, it starts a separate comment for the entire line. △

Some options can be on (enabled) or off (disabled). Specifying only the keyword enables the option, and specifying the keyword prefixed with NO disables the option. For example, a configuration file might contain the following option specifications to disable the options:

```
NOCENTER
NOSTIMER
NOSTATS
```

Options that take a value must be specified in the following way:

```
option-name=value
```

For example, a configuration file might contain the following lines:

```
LINESIZE=80
PAGESIZE=60
```

Note: When you specify SAS system options in a configuration file, blank spaces are not permitted before or after an equal sign. △

A configuration file can contain the CONFIG= option. A CONFIG= option in a configuration file can name a single ddname, data set name, or UFS filename. It cannot name a list of config files. The contents of the named config file are logically inserted in place of the CONFIG= specification. If a CONFIG= option specifies a file that has already been read as a configuration file, a warning message is written to the log and the file is not read again during this session.

The configuration file is processed as if all of the lines (other than comments) were concatenated into a single string with one blank space separating the lines, which means that options whose value can contain blank spaces can be continued across line boundaries. For example, the specification of the option in the following example is on five separate lines, but it would be processed as if it is on one line:

```
jreoptions=(
   -jreoption1
   -jreoption2
   -jreoption3
  )
```

In cases where separating concatenated lines with a blank space is not suitable, two alternative methods of explicit concatenation are provided.

- If the file has the legacy format, and there is a non-blank character in column 72, then the next line is concatenated without an intervening blank space. The character in column 72 is not ignored, it is included in the concatenated value.
- If the legacy method of explicit concatenation does not apply, and the last non-blank character of the line (or of columns 1-71 in a legacy format file) is a hyphen (-), then the hyphen is deleted and the next non-comment line is concatenated without an intervening blank space. If the last non-blank character is a plus sign (+), then the next non-comment line is concatenated without an intervening blank space, and any leading blank spaces of that line are also removed.

For example, the following option specification is invalid because a blank space is inserted between the equal sign and the value.

```
YEARCUTOFF=
1950
```

The following option specification is valid because the value is concatenated immediately following the equal sign, and a blank space is not inserted.

```
YEARCUTOFF=+
1950
```

Specifying a User Configuration File

To tell SAS where to find your user configuration file, do the following:

- If you use the SAS CLIST or SASRX exec to invoke SAS under TSO, use the CONFIG operand - for example:

  ```
  sas config(''my.config.file''')
  ```

  or

  ```
  sasrx -config 'my.config.file'
  ```

- If you use the SAS cataloged procedure to invoke SAS in batch mode, use the CONFIG= parameter - for example:

  ```
  //S1  EXEC SAS,CONFIG='MY.CONFIG.FILE'
  ```

The user configuration file that you specify is executed along with the system configuration file that your installation uses. This happens because the SAS CLIST, the SASRX exec, or the SAS cataloged procedure concatenates the file that you specified to the system configuration file.

During initialization, the specified value of the CONFIG= option is replaced with the list of all configuration files that are actually processed. PROC OPTIONS displays this new value.

Note: SAS system options that you specify in the user configuration file override system options that are specified in the system configuration file. △

Autoexec Files

Overview of Autoexec Files

Under z/OS, an *autoexec file* can be a sequential data set, a member of a partitioned data set, or a UFS file. Unlike configuration files, which contain SAS system options, an autoexec file contains SAS statements. These statements are executed immediately after SAS has been fully initialized and before any SAS input source statements have been processed. For example, an autoexec file could contain the following lines:

```
options fullstats pagesize=60 linesize=80;
libname mylib 'userid.my.lib';
dm 'clock';
```

The OPTIONS statement sets some SAS system options, the LIBNAME statement assigns a library, and the DM statement executes a command.

Note: Some SAS system options can be specified only when you invoke SAS. These system options cannot be specified in an OPTIONS statement; therefore, they cannot be

specified in an autoexec file. See Table 18.4 on page 477 for information about SAS system options and where they can be specified. △

Displaying Autoexec Statements in the SAS Log

SAS statements that are submitted from an autoexec file usually are not displayed in the SAS log. However, if you specify the ECHOAUTO system option when you invoke SAS, then SAS writes (or "echoes") the autoexec statements to the SAS log as they are executed.

Using an Autoexec File under TSO

Under TSO, use the AUTOEXEC operand when you invoke SAS to tell SAS where to find your autoexec file. For example, the following command invokes SAS and tells SAS to use an autoexec file named MY.EXEC.FILE:

```
sas autoexec(''my.exec.file''')
```

Using an Autoexec File in Batch Mode

To specify an autoexec file in a batch job, use a JCL DD statement to assign the ddname SASEXEC to your autoexec file. This DD statement must follow the JCL EXEC statement that invokes the SAS cataloged procedure. For example, the following two lines of JCL can be used to accomplish the same results in a batch job as the previous example did under TSO:

```
//MYJOB    EXEC SAS
//SASEXEC  DD DSN=MY.EXEC.FILE,DISP=SHR
```

SASUSER Library

Overview of the SASUSER Library

SAS enables you to customize certain features while your SAS session is running and to save these changes. The SASUSER library contains various SAS files in which SAS records these settings. For example, in Base SAS software, any changes that you make to function key settings or to window attributes are stored in a catalog named SASUSER.PROFILE. The SASUSER library can also contain personal catalogs for other SAS software products. You can also store SAS data files, SAS data views, SAS programs, SAS/ACCESS descriptor files, and additional SAS catalogs in your SASUSER library. In addition to storing function key settings and window attributes, the SASUSER.PROFILE catalog is used to store your DEFAULT.FORM. The DEFAULT.FORM is created by the FORM subsystem. It is used to control the default destination of all output that is generated by the PRINT command. For information about the FORM subsystem, see “Using the PRINT Command and the FORM Subsystem” on page 130 and the *SAS Language Reference: Dictionary*.

You can use three methods to set up your SASUSER library:

- Establish a permanent SASUSER library that is accessed for read and update. This method is how SASUSER must be set up if you want settings that are modified in the current SAS session to be in effect for a subsequent SAS session, which is the typical arrangement when you run SAS interactively. Creating a permanent SASUSER library is also necessary if you intend to save other application files in the SASUSER library for use in a later session. When accessing

a permanent SASUSER library for read and update, only one SAS session at a time can use the SASUSER library. To access a personal SASUSER library for read and update, leave the SASUSER option unspecified and allocate the ddname SASUSER to the SASUSER library. The SAS CLIST and SASRX exec that are supplied by SAS to invoke SAS under TSO work this way by default. They also create the SASUSER library data set if it does not exist. The default data set name that they use is **<prefix>.SAS9.SASUSER**, where **<prefix>** is the system prefix that is defined in your user profile, or with your user ID if no prefix is defined.

- Establish a permanent SASUSER library that is enabled for read access only. This method allows you to create a single SASUSER library that is shared by multiple SAS sessions that are running simultaneously. This method also allows you to provide other users with a SASUSER library that contains a set of pre-configured settings that are protected from write access via system authorization facilities (for a bound library) or via UFS permissions (for a UFS library). To access a SASUSER library in either a shared or read-only manner, you must specify the RSASUSER option. For more information about RSASUSER, see "RSASUSER System Option" in the *SAS Language Reference: Dictionary*.
- Establish a temporary SASUSER library that exists only for the lifetime of the current session. This method is appropriate for applications that can use the default settings and that do not need to save settings. If you do not specify the SASUSER option and do not allocate the SASUSER ddname, then SAS uses this method in the batch environment by default. You can also run interactive sessions in this manner by specifying the SASRX option NOSASUSER. When you do not specify a SASUSER library, SAS creates a new PROFILE catalog that is used to store profile information for use during the current SAS session. This catalog is placed in the WORK library, and a note to this effect is written to the SAS log. The WORK library is typically deleted at the end of your session, which means that any changes made to the PROFILE catalog will not be available in a subsequent SAS session.

If you are running SAS under TSO and you want to specify SASUSER in your SAS config file, then you need to specify the SASRX option NOSASUSER to prevent SASRX from allocating the SASUSER library before the config file is processed. This specification is available only if you are using SASRX. It is not available if you are using the CLIST because the CLIST does not have the NOSASUSER option.

Creating Your Own SASUSER Libraries

By creating your own SASUSER libraries, you can customize SAS software to meet the requirements of a number of different types of jobs. For example, suppose you want to create a user profile for a particular type of task that requires a unique set of key definitions.

To create this user profile, you must first create a SAS library that can be used as the SASUSER library. The easiest way to create this library is to start a SAS session and then use a LIBNAME statement to create the library, as explained in "Assigning SAS Libraries Internally" on page 70. For example, to create a SAS library with a physical filename of ABC.MY.SASUSER, submit the following LIBNAME statement:

```
libname newlib 'abc.my.sasuser' disp=new;
```

Notice that a libref of NEWLIB was used in this example. SASUSER is a reserved libref and cannot be reassigned during a SAS session.

You can also use the TSO ALLOCATE command to create a physical file for use as your SASUSER library. By using the ALLOCATE command, you can avoid using the LIBNAME statement; however, to use the ALLOCATE command effectively, you must be familiar with TSO commands and with DCB (data control block) attributes. Here is

a typical ALLOCATE command for the SASUSER library that provides satisfactory performance at many sites:

```
alloc fi(newlib) da('abc.my.sasuser') new
      catalog space(80 20) dsorg(ps) recfm(f s)
      blksize(6144) reu
```

When you enter this ALLOCATE command from the READY prompt, a physical file named ABC.MY.SASUSER is created with the correct attributes for a SAS library.

To use the new SAS library as the SASUSER library, you must end your SAS session and start a second session. When you start a second session, you can use the SASUSER option of the SAS CLIST or SASRX exec to specify ABC.MY.SASUSER as the SASUSER library.

Specifying Your Own SASUSER Library

After creating your own permanent SAS library, designate that library as your SASUSER library. You can do this in either of the following ways:

- Use the SASUSER option of the SAS CLIST or SASRX exec to specify the physical filename of your SAS library. For example, if you create a library with a name of ABC.MY.SASUSER, then you use the following CLIST command to invoke SAS:

  ```
  sas sasuser(''abc.my.sasuser''')
  ```

  Or, you would use the following SASRX command to invoke SAS:

  ```
  sasrx -sasuser 'abc.my.sasuser'
  ```

  When you enter this command, the libref SASUSER is associated with the SAS library whose physical filename is ABC.MY.SASUSER. Any profile changes that you make during your session are saved in the SAS catalog SASUSER.PROFILE, which is a member of the SASUSER library. These changes will be retained when you end your SAS session.
- Use the SASUSER= system option to specify the ddname that identifies your SAS library. (See "SASUSER= System Option" on page 588.)

Both of these methods require that you identify the SAS library when you invoke SAS; you cannot change the SASUSER library during a SAS session.

SAS System Options

Overview of SAS System Options

SAS system options control many aspects of your SAS session, including output destinations, the efficiency of program execution, and the attributes of SAS files and libraries.

After a system option is set, it affects all subsequent DATA and PROC steps in a process until it is specified again with a different value. For example, the CENTER|NOCENTER option affects all output from a process, regardless of the number of steps in the process.

Specifying or Changing System Option Settings

The default values for SAS system options are appropriate for many of your SAS programs. If you need to specify or change the value of a system option, you can do so in the following ways:

- Create a user configuration file to specify values for the SAS system options whose default values you want to override. See "Creating a User Configuration File" on page 9 for details.
- Under TSO, specify any SAS system option following the OPTIONS parameter in the SAS CLIST command:

  ```
  sas options('option-list')
  ```

  For options that can be on or off, just list the keyword that corresponds to the appropriate setting. For options that take a value, list the keyword identifying the option followed by an equal sign and the option value, as in the following example:

  ```
  sas options('nodate config=myconfig')
  ```

  See Appendix 1, "Starting SAS with SASRX," on page 641 for detailed information about the SASRX exec.
- In batch mode, specify any SAS system option in the EXEC SAS statement:

  ```
  // EXEC SAS,OPTIONS='option-list'
  ```

  For example:

  ```
  // EXEC SAS,OPTIONS='OPLIST LS=80 NOSTATS'
  ```

- Specify SAS system options in an OPTIONS statement in an autoexec file, which is executed when you invoke SAS, or in an OPTIONS statement at any point during a SAS session. Options specified in an OPTIONS statement apply to the process in which they are specified, and are reset for the duration of the SAS session or until you change them with another OPTIONS statement.

  For example:

  ```
  options nodate linesize=72;
  ```

  See Table 18.4 on page 477 to find out whether a particular option can be specified in the OPTIONS statement. For more information about autoexec files, see "Autoexec Files" on page 11. For more information about the OPTIONS statement, see *SAS Language Reference: Dictionary* and *Step-by-Step Programming with Base SAS Software*.
- Change SAS system options from within the OPTIONS window. On a command line, enter the keyword OPTIONS. The OPTIONS window appears. Place the cursor on any option setting and type over the existing value. The value is saved for the duration of the SAS session only. Not all options are listed in the OPTIONS window. See "OPTIONS Window" on page 17 for more information.
- Specify PROC OPTLOAD or the DMOPTLOAD command to load a set of options that you have saved in a file or data set by using PROC OPTSAVE or the DMOPTSAVE command. For example, specifying

  ```
  proc optload data=options1;
  run;
  ```

  loads the set of options that you have saved in a file named **options1**. You can save multiple sets of options, and then use the OPTLOAD procedure to load any of your sets of options at any time during a SAS session. The ability to load the options at any time during a SAS session provides advantages over using a configuration file, which you can use only when you invoke SAS. However, not all options are saved by PROC OPTSAVE. For information about which options cannot be saved with PROC OPTSAVE, see "The OPTSAVE Procedure" .

Determining How an Option Was Set

Because of the relationship between some SAS system options, SAS might modify an option's value. This modification might change your results.

To determine how an option was set, enter the following code in the SAS Program Editor:

```
proc options option=option value; run;
```

After you submit this code, the SAS log will display the value that was set for the option and how the value was set. For example, the following log message is displayed when you enter

```
proc options option=CATCACHE value; run;
```

Output 1.1 Results of the OPTIONS Procedure for the CATCACHE Option

```
Option Value Information for SAS Option CATCACHE
      Option Value:  0
      Option Scope:  NoReb
      How option value was set:  Shipped Default
```

Contact your on-site SAS support personnel for more information.

Default Options Table and Restricted Options Table

Your on-site SAS support personnel might have created a default options table or a restricted options table. information about creating and maintaining these tables is provided in the configuration guide for SAS software in the z/OS environment.

The default options table is created by your site administrator and provides a global set of defaults that are specific to your site. It reduces the need to duplicate options in every system config file at your site.

The restricted options table is created by the site administrator. It specifies option values that are established at startup and cannot be overridden. If an option is listed in the restricted options table, any attempt to set it is ignored. An attempt to set it with the options statement causes a warning message to be written to the log. For example

```
1    options vsamload;
             --------
             36
WARNING 36-12: SAS option VSAMLOAD is restricted by your Site Administrator and
cannot be updated.
```

To find out which options are in the restricted options table, submit this statement:

```
PROC OPTIONS RESTRICT;
RUN;
```

For a list of restricted options, see "Restricted Options" in *SAS Language Reference: Dictionary*.

Some SAS system options cannot be added to a restricted options table. To find out whether an option can be restricted, run PROC OPTIONS with the DEFINE option. For example:

```
PROC OPTIONS OPTION=option-name DEFINE;
RUN;
```

The output that is returned by the preceding statement will include either of the following messages:

```
Your Site Administrator can restrict modification of this option.
```

or

```
Your Site Administrator cannot restrict modification of this option.
```

If an ineligible option has been placed in the restricted options table, the following message is issued:

```
SAS option option-name cannot be restricted by your Systems Administrator.
```

SAS terminates with an abend. If you receive such an error, you should immediately notify your site administrator.

Displaying System Option Settings

To display the current settings of SAS system options, use the OPTIONS procedure or the OPTIONS window.

Some options might seem to have default values even though the default value listed in Table 18.4 on page 477 is none. This happens when the option is set in a system configuration file, in the default options table, or in the restricted options table.

You can use the VALUE parameter of the OPTIONS procedure to see when an option's value was set.

OPTIONS Procedure

The OPTIONS procedure writes system options that are available under z/OS to the SAS log. By default, the procedure lists one option per line with a brief explanation of what the option does. To list the options with no explanation, use the SHORT option:

```
proc options short;
run;
```

To list all the options in a certain category, use the GROUP= option:

```
proc options group=sort;
run;
```

Some options, such as system options that are specific to SAS/ACCESS interfaces or to the SAS interface to ISPF, are listed only if you specify the GROUP= option. See "OPTIONS Procedure" on page 371 for details.

OPTIONS Window

To display the OPTIONS window, enter **OPTIONS** on a command line. The OPTIONS window displays the settings of many SAS system options.

Precedence for Option Specifications

When the same option is set in more than one place, the order of precedence is as follows:

1. restricted options table, if there is one
2. OPTIONS statement or OPTIONS window
3. SAS invocation, including invocation by way of an EXEC SAS JCL statement (in batch) or by way of the SAS CLIST or SASRX exec commands (under TSO)
4. user configuration file, if there is one
5. system configuration file (as SAS software is initialized)
6. default options table, if there is one

For example, options that you specify during your SAS session (using the OPTIONS statement or OPTIONS window) take precedence over options that you specified when you invoked SAS. Options that you specify with the SAS CLIST or SASRX exec commands take precedence over settings in the configuration file. The settings in the user configuration file take precedence over settings in the system configuration file and in the default options table.

Note: Options that are specified in the restricted options table can be updated only by your SAS administrator. △

HFS, UFS, and zFS Terminology

Historically, the term HFS was used to refer to UNIX file systems in general on z/OS as well as to a particular physical file system implementation. However, since IBM has introduced zFS as a functional replacement for the HFS physical file system, the term HFS can be misleading. Therefore, SAS documentation now uses the term UFS to refer to UNIX file systems in general. The terms HFS, zFS, and NFS designate three different physical file system implementations. Except in rare cases, SAS processing for UFS files and directories is identical, regardless of the physical file system implementation.

Your systems administrator, not SAS, controls whether the HFS or zFS implementation is used for a particular file system.

Most occurrences of HFS in the documentation have been changed to UFS. HFS is still used in feature names and in syntax statements and prefixes where it is the correct term. UFS cannot be substituted for HFS in these contexts. The following table lists the terminology changes that have been made.

Table 1.2 Terminology Changes

Old Term	New Term
HFS file system	UFS file system
HFS file	UFS file
HFS library	UFS library

For more information about the UFS file system, UFS libraries, and UFS files, see “UFS Libraries” on page 61.

Specifying Physical Files

Overview of Physical Files

Wherever you specify the name of a physical file internally (for example, in a SAS LIBNAME or FILENAME statement, in a LIBNAME or FILENAME function, in a DATA step, or in a SAS procedure), the name can be in any of these forms:

- a fully qualified data set name such as ‘SAS.SAS9.AUTOEXEC’. A PDS member name, such as ‘MY.PDS(MEMBER)’, might also be specified, although not for a LIBNAME statement or function.

- a partially qualified data set name such as".CNTL'. SAS inserts the value of the SYSPREF= system option (which is usually *user ID* by default) in front of the period. (See "SYSPREF= System Option" on page 613.) In the following example, an OPTIONS statement is used to assign a value of USER12.SAS9 to the SYSPREF= system option. When SAS executes the FILENAME statement, it interprets '.RAW.DATAX' as 'USER12.SAS9.RAW.DATAX'.

```
options syspref=user12.sas9;
filename raw2 '.raw.datax' disp=old;
```

- a temporary data set name such as '&MYTEMP'.
- a UFS path. It can be a full path that begins with a slash (/) or a tilde (~), as the following examples indicate:

```
filename fullpath '~/subdir/filename.sas';
filename relative 'subdir/filename.sas';
```

- a concatenated series of names or a wildcard name consisting of multiple UNIX File System (UFS) files or members of a partitioned data set (PDS, PDSE). See "Concatenating External Files" on page 92. However, note that the LIBNAME statement and LIBNAME function does not support the wildcard syntax for members of partitioned data sets. It is possible to concatenate SAS libraries. For details, see the LIBNAME statement "LIBNAME Statement" on page 451.

Note that names of physical files should be enclosed in quotation marks.

Specifying Physical Files with the INCLUDE Command

Here are examples of the INCLUDE command that illustrate the various ways you can specify physical files:

INCLUDE MYPGM
MYPGM is a fileref that was previously associated with an external file.

INCLUDE MYPGM(PGM1)
PGM1 is a member of the partitioned data set that is associated with the fileref MYPGM.

INCLUDE 'USERID.TEST.PGMS'
This is an example of a sequential data set name.

INCLUDE 'MVS:USERID.TEST.PGMS'
This is an example of a sequential data set name that uses the designator for the MVS file system.

INCLUDE 'USERID.TEST.PGMS(AAA)'
This is an example of a data set name with a member specified.

INCLUDE '.TEST.MYPGM'
Assuming that the FILESYSTEM= system option is set to MVS, SAS adds a prefix to this data set name that contains the value of the SAS system option SYSPREF=, which defaults to your system prefix. If FILESYSTEM=HFS, SAS looks into your default UNIX System Services directory for the "hidden" file .TEST.MYPGM.

INCLUDE 'HFS:/u/userid/mypgms/mypgm1.c'
This is an example of a path to a UNIX File System (UFS) file in the Hierarchical File System (HFS), represented by a fully qualified path. For more information about HFS and UFS, see "HFS, UFS, and zFS Terminology" on page 18.

INCLUDE 'pgms/mypgms/mypgm1.c'
This is an example of a relative path to a UNIX File System file. Any filename containing a slash (/) is assumed to be in UNIX File System, regardless of the value of the FILESYSTEM= system option. If the pathname does not begin with a slash (/), then SAS searches for the file in the default UFS directory for that user.

Handling of Nonstandard Member Names

You can use the SAS system option FILEEXT= to specify how extensions in member names of partitioned data sets are to be handled. See "FILEEXT= System Option" on page 520 for more information.

SAS Software Files

Overview of SAS Software Files

Configuration files (described in "Configuration Files" on page 9) and SASUSER files (described in "SASUSER Library" on page 12) are only two of several SAS software files that are automatically identified to your session by either the SAS CLIST or SASRX exec (under TSO) or the SAS cataloged procedure (in batch). This section describes several other SAS software files that are significant to SAS users under z/OS.

For brief descriptions of all the SAS software files that are frequently used by the SAS CLIST, the SASRX exec, or by the SAS cataloged procedure, see Table 1.4 on page 29.

WORK Library and Other Utility Files

Overview of the Work Library

The WORK library is a special-purpose SAS library that contains temporary files, including certain types of utility files that are created by SAS as part of processing the current SAS session or job. The WORK library is also the default location for one-level member names and user settings. One-level member names are member names that are specified without a libref. They reside in the WORK library unless a USER library has been identified. See "USER Library" on page 23 for more information about USER libraries. In a single-user SAS session or job, if the SASUSER library is not assigned, the WORK library also houses temporary user settings in files, such as the PROFILE catalog and the SAS registry itemstore.

In a single-user SAS session or job, the WORK library is typically created at the beginning of a SAS session or job and deleted at the end. Multi-user SAS servers also create a WORK library (referred to as a *client WORK library*) for each client that connects to the server. Client WORK libraries contain the temporary files created as part of the processing done by the server on behalf of the client. The server creates a distinct client WORK library for each client so that files used by one client are not commingled with files belonging to another client. These libraries are created when the client establishes a connection with the server, and they exist until the client disconnects.

The WORK library is always processed by the BASE engine, which requires that you must use one of the following library implementation types for WORK:

The advantages of each of these library implementation types for use with the WORK library are detailed in the following topics, along with usage notes. See "Library Implementation Types for Base and Sequential Engines" on page 52 for information about each type of library.

Note: For multi-user SAS servers, each client WORK library that is created has the same implementation type as the WORK library for the SAS server that is specified at server initialization. △

Direct Access Bound Library

Assigning a direct access bound library for the WORK library is generally the best choice for single-user SAS sessions or jobs. At most installations, the default WORK library allocation provided by the SASRX exec, SAS CLIST, or JCL procedure is appropriate for a wide class of jobs. Users can usually increase the allocation to provide additional space without assistance from a systems administrator.

Direct access bound libraries have a maximum size to which they can grow. If the WORK library becomes full, an attempt to create or extend a member will fail, which usually results in a message similar to this one:

```
ERROR: Write to WORK <member>.<type> failed. File is full and might be damaged.
```

In most cases, SAS processing cannot succeed if the WORK library becomes full. However, allocating an excessive amount of space for the WORK library ties up disk space that could be used by other jobs. Therefore, some advance planning is often necessary for large SAS jobs to ensure that the appropriate amount of space is allocated to the WORK library. Often, the best way to determine whether you have enough space is to run the SAS job or session to perform a typical processing job, and then measure the amount of WORK library space that the job uses. You can measure the required space by including the following statement at the end of your job:

```
proc datasets lib=work; quit; run;
```

In the information that is listed for the library directory, the statistic "Highest Formatted Block" represents the largest number of blocks in simultaneous use at any time during the SAS session. Divide this statistic by the "Blocks per Track" statistic to convert it to the maximum number of disk tracks required for the WORK library during this session. Using this information, you can derive the primary and secondary space allocation for the WORK library data set with the following method:

1. Select a primary allocation which is approximately equivalent to the minimum expected space utilization.
2. Select a secondary allocation that allows sufficient growth up to the maximum expected space utilization.

See "General Guidelines" on page 78 for more information.

Note:

□ Some SAS procedures write utility files to the WORK library only if the NOTHREADS system option has been specified. Therefore, a smaller amount of WORK library space might be required if you specify THREADS.

□ For processing performed by a multi-user SAS server, the libref WORK refers to the client WORK library only in certain contexts, such as in SAS code submitted for execution by the server. See the documentation for the specific SAS server to determine how to access the client WORK library.

- Allocating the WORK library data set to VIO (virtual I/O) can avoid physical I/O and thus decrease the elapsed time required for a SAS job to run. Therefore, VIO is often preferable to actual disc devices, especially for jobs with modest WORK space requirements. To use VIO, a particular UNIT name typically must be specified with the SASRX exec, SAS CLIST, or JCL procedure. For assistance, contact your z/OS systems administrator. See "Consider Designating Temporary SAS Libraries as Virtual I/O Data Sets" on page 630 for a list of restrictions for using VIO. If you are using a SAS CLIST, a SASRX exec, or JCL procedure to invoke SAS, contact your systems administrator for information about how to control the allocation of the DDNAME WORK.

△

The size of the WORK library for a single-user SAS session or job is controlled by the space parameters that you specify on the allocation of the DDNAME WORK. See Appendix 1, "Starting SAS with SASRX," on page 641 for the WORK option that you use to specify the primary and secondary space allocations for WORK.

When WORK resides in a direct access bound library, the size of the client WORK library is controlled with the values of the following SAS system options that are specified when the SAS server environment is initialized:

FILESPPRI	primary space allocation
FILESPSEC	secondary space allocation
FILEUNIT	unit of space

These same values are used for creating each client WORK library; therefore, it is necessary to specify values that are appropriate for all client processing that is performed by the server. This limitation can be avoided by using a UFS library for WORK.

UFS Library

Placing the WORK library in a UFS directory eliminates the need to specify the amount of space which is allocated to the WORK library (including client WORK libraries). This feature is particularly valuable for multi-user SAS servers because the space requirements for individual client WORK libraries might vary widely and be difficult to predict. When you place your WORK library in a UFS directory, each WORK library uses only the space it actually needs for the files that are created, and this space is drawn from a large pool. A pool consists of the free space available in the UFS file system in which the directory is located.

To use UFS libraries, follow these guidelines:

- Specify the path to the directory in which the WORK library (or libraries) will reside with the SAS system option WORK, as shown in the following example:

```
WORK="/tmp"
```

  Each WORK library (or client WORK library) will reside in a subdirectory within the UFS directory that you specify with the WORK option. These subdirectories are created automatically as they are needed.

  SAS recommends that the specified UFS path correspond to a directory, such as **/tmp**, that has its sticky bit turned on. When the sticky bit is on for a directory, directories that are contained within that directory can be removed only by the owner of the directory, the owner of the directory that is being deleted, or by a superuser. This setting allows multiple SAS users to place temporary directories in the same location without the risk of accidentally deleting each other's files.

- If necessary, specify the SAS system option WORKTERM to remove the WORK library subdirectories and their contents when they are no longer needed. For more information about using WORKTERM, see "WORKTERM System Option" on page 621.
- Avoid allocating the DDNAME WORK (if possible), or allocate the minimum amount of space, because the data set allocated to the DDNAME WORK is not used when a UFS path is specified for the WORK option.

For more information about UFS, see "HFS, UFS, and zFS Terminology" on page 18.

Hiperspace Library

Hiperspace libraries reside in memory, so they can be used to avoid I/O for WORK library processing. Using a hiperspace library reduces the elapsed time that is required for SAS processing that uses the WORK library. In a multi-user SAS server environment, the space for all client WORK libraries is drawn from a single pool for the server that is shared by all of the clients that are using the server. This pool provides more flexibility in space allocation than when you use direct access bound libraries.

Note: Some installations might place limits on hiperspace use. Consequently, using hiperspace for WORK might be more appropriate for SAS jobs or servers with modest WORK space requirements. Contact your systems administrator for information about hiperspace limitations. △

To use hiperspace libraries, follow these guidelines:

- Specify the SAS system option "HSWORK System Option" on page 546.
- Avoid allocating the DDNAME WORK (if possible), or allocate the minimum amount of space, because the data set allocated to the DDNAME WORK is not used when HSWORK is specified.
- Specify the SAS system option NOHSSAVE to avoid I/O unless you are using DIV libraries, which require updates to be committed immediately.
- Specify values that provide sufficient total space for the entire SAS job or server for the following SAS system options:
  - "HSLXTNTS= System Option" on page 543
  - "HSMAXPGS= System Option" on page 544
  - "HSMAXSPC= System Option" on page 544

USER Library

You can identify a permanent library in which SAS will store members specified with one-level names (that is, without a libref). This feature can be useful for applications that require a default location for SAS files that is permanent or that exists beyond the end of the current SAS session.

To use a USER library, follow these guidelines:

- Assign a libref to the user library data set.
- Specify the libref as a value of the "USER= System Option" on page 615.

Utility Files That Do Not Reside in WORK

In SAS®9, some SAS procedures create a new type of utility file that does not reside in WORK but rather in a location specified via the UTILLOC system option. In some cases, these utility files are created only if the THREADS system option is set to a

nonzero value. These utility files, which provide certain performance benefits, can reside in one of two different types of locations on z/OS:

temporary z/OS data set
: Each utility file resides in a separate, temporary, sequential data set on disk (or VIO) that has a system-generated name that is allocated by a system-generated DDNAME. The amount of disk space available to each utility file is specified by an ALLOC command, which is specified as the value of the UTILLOC option. Specify the UCOUNT keyword to allow these files to reside in multi-volume data sets.

UFS file
: Each utility file is a UFS file residing in a temporary directory subordinate to the UFS path specified for the UTILLOC option.

See “UTILLOC= System Option” on page 616 for more information about the UTILLOC system option and the UCOUNT keyword.

SAS Log File

Overview of the SAS Log File

The SAS log file is a temporary physical file that has a ddname of SASLOG in the SAS cataloged procedure, the SAS CLIST, and the SASRX exec. In batch mode, the SAS cataloged procedure assigns default data control block (DCB) characteristics to this file as follows:

BLKSIZE=141

LRECL=137

RECFM=VBA

Under TSO, either interactively or noninteractively, the SASLOG file is routed to the terminal by default. In the windowing environment, the SAS log is directed to the Log window.

See “Types of SAS Output” on page 120 for more information about the SAS log and about how to route output in a batch job.

Changing the Contents of the SAS Log

The particular information that appears in the SAS log depends on the settings of several SAS system options. See “Collecting Performance Statistics” on page 626 for more information.

In addition, the following portable system options affect the contents of the SAS log:

CPUID
: controls whether CPU information is printed at the beginning of the SAS log.

DETAILS
: specifies whether to include additional information when files are listed in a SAS library.

ECHOAUTO
: controls whether the SAS source statements in the autoexec file are written (echoed) to the SAS log.

MLOGIC
: controls whether macro trace information is written to the SAS log when macros are executed.

MPRINT
controls whether SAS statements that are generated by macros are displayed.

MSGLEVEL
controls the level of messages that are displayed.

NEWS=
specifies an external file that contains messages to be written to the SAS log when SAS software is initialized. Typically, the file contains information such as news items about the system.

NOTES
controls whether NOTES are printed in the log. NOTES is the default setting for all methods of running SAS. Do not specify NONOTES unless your SAS program is completely debugged.

OPLIST
specifies whether options given at SAS invocation are written to the SAS log.

PAGESIZE=
specifies the number of lines that compose a page of SAS output.

PRINTMSGLIST
controls whether extended lists of messages are printed.

SOURCE
controls whether SAS source statements are written to the log. NOSOURCE is the default setting for SAS interactive line mode; otherwise, SOURCE is the default.

SOURCE2
controls whether secondary source statements from files that are included by %INCLUDE statements are written to the SAS log.

SYMBOLGEN
controls whether the macro processor displays the results of resolving macro references.

Changing the Appearance of the SAS Log

The following portable system options are used to change the appearance of the SAS log:

DATE
controls whether the date and time, based on when the SAS job or session began, are written at the top of each page of the SAS log and of any print file that SAS software creates. Use NODATE to suppress printing of the date and time.

LINESIZE=
specifies the line size (printer line width) for the SAS log and the SAS procedure output file. LS= is an alias for this option. LINESIZE= values can range from 64 through 256.

NUMBER
controls whether the log pages are numbered. NUMBER is the default. Use the NONUMBER option to suppress page numbers.

OVP
controls whether lines in SAS output are overprinted.

SAS Procedure Output File

Overview of the SAS Procedure Output File

Whenever a SAS program executes a PROC step that produces printed output, SAS sends the output to the procedure output file. Under TSO, either interactively or noninteractively, the procedure output file is routed to the terminal by default. In the windowing environment, output is directed to the Output window.

In batch mode, the SAS procedure output file is identified in the cataloged procedure by the ddname SASLIST. Unless you specify otherwise, SAS writes most procedure output to this file. (A few procedures, such as the OPTIONS procedure, route output directly to the SAS log by default.) PUT statement output might also be directed to this file by a FILE statement that uses the fileref PRINT. (PRINT is a special fileref that can be specified in the FILE statement.)

The following DCB characteristics of the procedure output file are controlled by the cataloged procedure, typically with the following values:

BLKSIZE=264

LRECL=260

RECFM=VBA

The SAS procedure output file is often called the *print file*; however, any data set that contains carriage-control information (identified by a trailing A as part of the RECFM= specification) can be called a print file.

Changing the Appearance of Procedure Output

The following portable system options are used to change the appearance of procedure output:

CENTER
controls whether the printed results are centered or left-aligned on the procedure output page. CENTER is the default; NOCENTER specifies left alignment.

DATE
controls whether the date and time, based on when the SAS job or session began, are written at the top of each page of the SAS log and of any print file that SAS software creates. Use NODATE to suppress printing of the date and time.

LINESIZE=
specifies the line size (printer line width) for the SAS log and the SAS procedure output file. LS= is an alias for this option. LINESIZE= values can range from 64 through 256.

NUMBER
controls whether the page number is printed on the first title line of each SAS printed output page. NUMBER is the default. Use the NONUMBER option to suppress page numbers.

PAGENO=
specifies a beginning page number for the next page of output that SAS software produces.

PAGESIZE=
specifies how many lines to print on each page of SAS output. PS= is an alias for this option. In the windowing environment or in an interactive line mode session, the PAGESIZE= option defaults to the terminal screen size, if this information is

available from the operating environment. PAGESIZE= values can range from 15 through 500.

Console Log File

The SAS console log file is a physical file that is automatically allocated at the start of SAS initialization. The console log file records log messages generated when the regular SAS log is either unavailable or is not yet initialized. You can control the appearance of the console log file with the LINESIZE= system option only. The SAS CLIST, the SASRX exec, and cataloged procedures allocate this file using the ddname SASCLOG.

Parmcards File

The parmcards file is a temporary physical file that is identified by the ddname SASPARM. It is created automatically by the SAS cataloged procedure and by the SAS CLIST or SASRX exec. SAS uses the parmcards file for internal processing. Lines that follow a PARMCARDS statement in a PROC step are first written to the parmcards file; then they are read into the procedure. The PARMCARDS statement is used in the BMDP and EXPLODE procedures.

TKMVSENV File

A TKMVSENV file is created at install time. You can use the ddname TKMVSENV with SAS procedures, the SASRX exec, and CLISTs to point to the file. The file must be a sequential file or a member of a PDS with a record format of fixed blocked.

The TKMVSENV file is used to make a list of pseudo environment variables, which are available to SAS Scalable Architecture applications. See *SAS Scalable Performance Data Server: User's Guide* and *SAS Scalable Performance Data Engine: Reference* for more information about SAS Scalable Architecture. Environment variables are supported for your SAS administrator to use to tailor applications that use SAS Scalable Architecture. Some environment variables are used by SAS Technical Support to investigate problems that are reported by users.

Each record in the TKMVSENV file must contain a single command: SET or RESET. The RESET command clears all previously set environment variables. The SET name=*value* command allows you to create the variable *name* and assign it the value *value*.

Each command must begin in column 1 of the record. No blank spaces are permitted in the name=*value* specification on the SET command, except when the value can be enclosed in quotation marks. Some variables have a Boolean effect. These variables are turned on when they are defined and turned off when they are not defined. Such variables do not need to have a value and can be defined by using the SET name= command.

You can include comments after the command specification by adding one or more blank spaces between the command specification and the comment. SAS 9.2 enables you to comment out entire records, as well as add comments after a command specification. Any record that has an asterisk in column 1 will be ignored, and the entire record will be treated as a comment.

set TKOPT_SVCNO=*nnn*
set TKOPT_SVCR15=*nn*
These variables tell the SAS Scalable Architecture interface how the SAS SVC is installed at the user site. This information is necessary because the SAS Scalable Architecture interface might need to use some of the SVC services independently

of the SAS application. These variables should be specified with the same values as the SAS options of the same name.

set TKOPT_NOHFS=
This Boolean variable is provided for those sites that are unable to provide basic UFS file system resources to SAS. If this option is specified, then the SAS Scalable Architecture interface takes the following action when a UFS file open is requested:

- If the file open request is an INPUT open request, the file is treated as an empty file. No UFS files are opened.
- If the file open request is an OUTPUT open request, a SYSOUT data set is allocated with a ddname of TKHFS*nnn*, where *nnn* is a unique number that is increased throughout the session. The first record in the SYSOUT data set contains the pathname of the UFS file actually requested. The remaining records contains the data intended for the named UFS file.

For more information about HFS and UFS, see "HFS, UFS, and zFS Terminology" on page 18

set TKOPT_LPANAME=*xxxxxxxx*
This option specifies the name of the SAS application entry point invoked by the SASLPA main entry point. If the installation placed the LPA resident module in an LPA with a name other than SASXAL, the user needs to specify the same name for the TKOPT_LPANAME option value.

set TKOPT_UMASK=*nnn*
This option specifies the UNIX **umask** to apply to this session. This mask is applied to any UFS files created and operates as a standard UNIX **umask**. *nnn* must be exactly three octal digits between 0 and 7.

set TKOPT_CWD=*path*
This option causes the current working directory to be set to *path* for the SAS session. If the pathname is nonexistent or invalid, no action is taken. The path can be absolute or relative.

set DISABLESASIPV6=
This Boolean variable disables support for TCP/IP IPv6 on z/OS.

set TCPIPMCH=xxxxxxxx
This option specifies the IBM TCP/IP stack name to set the stack affinity for z/OS systems that are running more than one TCP/IP stack.

set TCPRSLV=IBM | SASC
This option sets the TCP/IP DNS resolver to either the IBM DNS Resolver or to the SAS/C DNS Resolver. By default, SAS uses the IBM DNS Resolver unless the DISABLESASIPV6 option has been set.

For more details about the environment variables that are supported and their recommended values, see the following sources.

Table 1.3 SAS References

Type of environment variables	Reference
SAS Installation	*Configuration Guide for SAS 9.2 for z/OS*
SAS Troubleshooting	SAS Technical Support
Configuring for the Java Platform	*Configuration Guide for SAS 9.2 for z/OS*

Summary Table of SAS Software Files

Table 1.4 on page 29 lists all of the SAS software files that are frequently used in the SAS CLIST, the SASRX exec, or in the SAS cataloged procedure. In the CLIST, SASRX, and cataloged procedure, logical names are associated with physical files. The logical names listed in the table are those that are used by the standard SAS CLIST, SASRX, or cataloged procedure. Your installation might have changed these names.

The system option column in the table lists the SAS system options that you can pass into the SAS CLIST or SASRX (using the OPTIONS operand) or into the SAS cataloged procedure (using the OPTIONS parameter) when you invoke SAS. You can use these system options to change the defaults that were established by the CLIST, SASRX, or by the cataloged procedure. (See "Specifying or Changing System Option Settings" on page 14.)

Table 1.4 SAS Software Files

Default Logical Name	Purpose	System Option	CLIST Operands	Type of OS Data Set
CONFIG	system configuration file	CONFIG= *ddname*	DDCONFIG(*ddname*)	sequential data set or PDS member
Description: contains system options that are processed automatically when you invoke SAS. The system configuration file is usually maintained by your data center.				
CONFIG	user configuration file	CONFIG= *ddname*	CONFIG(*dsn*) DDCONFIG(*ddname*)	sequential data set or PDS member
Description: also contains system options that are processed automatically when you invoke SAS. Your user configuration file is concatenated to the system configuration file.				
LIBRARY	format library	not applicable	not applicable	SAS library
Description: contains formats and informats.				
SAMPSIO	sample SAS library	not applicable	not applicable	SAS library
Description: is the SAS library that is accessed by SAS programs in the sample library provided by SAS Institute.				
SAS*nnnnn*	command processor file	not applicable	not applicable	sequential data set or PDS member
Description: is used by the SASCP command in the SAS CLIST or the SASRX exec.				
SASAUTOS	system autocall library	not applicable	MAUTS(*dsn*)	PDS
Description: contains source for SAS macros that were written by your data center or provided by SAS Institute.				
SASAUTOS	user autocall library	SASAUTOS= *specification**	SASAUTOS(*dsn*) DDSASAUT(*ddname*)	PDS

Default Logical Name	Purpose	System Option	CLIST Operands	Type of OS Data Set
Description: contains a user-defined autocall library to which the system autocall library is concatenated.				
SASCLOG	console log	not applicable	not applicable	sequential data set or PDS member
Description: SAS console log file.				
SASEXEC	autoexec file	AUTOEXEC= *ddname*	AUTOEXEC(*dsn*) DDAUTOEX(*ddname*)	sequential data set or PDS member
Description: contains statements that are executed automatically when you invoke SAS.				
SASHELP	HELP library	SASHELP= *ddname*	SASHELP(*dsn*) DDSASHLP(*ddname*)	SAS library
Description: contains system default catalogs and Help system information.				
SASLIB	format library (V5)	SASLIB= *ddname*	not applicable	load library
Description: a load library that contains user-written procedures and functions or Version 5 formats and informats. It is searched before the SAS software load library.				
SASLIST	procedure output file	PRINT= *ddname*	PRINT(*dsn*) DDPRINT(*ddname*)	sequential data set or PDS member
Description: contains SAS procedure output.				
SASLOG	log file	LOG= *ddname*	LOG(*dsn*) DDLOG(*ddname*)	sequential data set or PDS member
Description: SAS log file.				
SASMSG	system message file	SASMSG= *ddname*	SASMSG(*dsn*) DDSASMSG(*ddname*)	PDS
Description: contains SAS software messages.				
SASPARM	parmcards file	PARMCARD= *ddname*	PARMCARD(*size*) DDPARMCD(*ddname*)	sequential data set or PDS member
Description: a temporary data set that is used by some procedures. The PARMCARD= system option assigns a ddname to the parmcards file; the PARMCARD CLIST or SASRX operand specifies the file size. You can use the DDPARMCD operand to specify an alternate name for the parmcards file via the CLIST or SASRX.				
SASSNAP	SNAP dump file	not applicable	not applicable	sequential data set or PDS member
Description: SNAP output from dump taken during abend recovery.				
SASSWK*nn*	sort work files	DYNALLOC SORTWKDD= SORTWKNO=	not applicable	sequential
Description: temporary files that are used by the host sort utility when sorting large amounts of data.				
SASUSER	SASUSER library	SASUSER= *ddname*	SASUSER(*dsn*) DDSASUSR(*ddname*)	SAS library
Description: contains the user profile catalog and other personal catalogs.				
STEPLIB	STEPLIB library	not applicable	LOAD(*dsn*) SASLOAD(*dsn*)	load library

Default Logical Name	Purpose	System Option	CLIST Operands	Type of OS Data Set
Description: a load library that contains SAS procedure and user-written load modules. (Allocate with a STEPLIB DD statement in a batch job.)				
SYSIN	primary input file	SYSIN= *ddname*	INPUT(*dsn*) DDSYSIN(*ddname*)	sequential data set or PDS member
Description: contains SAS statements. The primary input file can be specified with the INPUT operand under TSO, or allocated with a DD statement in a batch job.				
TKMVSENV	TKMVSENV file	not applicable	not applicable	sequential data set or PDS member
Description: contains a list of pseudo environment variables that are available to SAS Scalable Architecture applications.				
USER	USER library	USER= *ddname* | *dsn*	not applicable	SAS library
Description: specifies a SAS library in which to store SAS data sets that have one-level names (instead of storing them in the WORK library).				
WORK	WORK library	WORK= *ddname*	DDWORK(*ddname*)	SAS library
Description: contains temporary SAS files that are created by SAS software during your session.				

* SASAUTOS: *specification* can be a fileref, a partitioned data set name enclosed in quotation marks, or a series of file specifications enclosed in parentheses.

Transporting SAS Data Sets between Operating Environments

SAS supports three ways of transporting SAS data sets between z/OS and other SAS operating environments: the XPORT engine, the CPORT and CIMPORT procedures, and SAS/CONNECT software, which is licensed separately. The process of moving a SAS file to or from z/OS with the XPORT engine or with the CPORT and CIMPORT procedures involves three general steps:

1. Convert the SAS file to the intermediate form known as *transport format*.
2. Physically move the transport format file to the other operating environment.
3. Convert the transport format file into a normal, fully functional SAS file, in the format required by the other operating environment.

For further information about the XPORT engine and on the CPORT and CIMPORT procedures, including limited restrictions, refer to *Moving and Accessing SAS Files*.

SAS/CONNECT software allows you to move files between operating environments without using the intermediate transport format. For further information about SAS/CONNECT, including limited restrictions, refer to *Communications Access Methods for SAS/CONNECT and SAS/SHARE*.

Accessing SAS Files in Other Operating Environments

SAS supports read-only cross-environment data access (CEDA) for certain types of SAS files created in the format of SAS Version 7 or later. CEDA allows you to read files in other operating environments as if those files were stored under z/OS. For more

information about CEDA, see *Moving and Accessing SAS Files* and the information about the Migration focus area at **support.sas.com/migration**.

Using Input/Output Features

SAS 5 and SAS 6 data sets generally need to be migrated to SAS 9.2 to enable you to use the I/O features introduced in SAS 9 and SAS 8. For example, to add integrity constraints to a SAS 6 data set, you must first migrate that data set to SAS 9.2. For information about migrating your data sets, see the Migration focus area at **support.sas.com/migration**. For information about I/O features introduced in SAS 9, SAS 9.1, and SAS 9.2, refer to the *SAS Language Reference: Dictionary*.

Reserved z/OS ddnames

In addition to the logical names shown in Table 1.4 on page 29, which have a special meaning to SAS, you should be aware of the following reserved ddnames, which have a special meaning to the operating environment:

JOBCAT
: specifies a private catalog that the operating environment is to use instead of the system catalog for the duration of the job (including jobs with more than one job step).

JOBLIB
: performs the same function as STEPLIB (described in Table 1.4 on page 29) except that it can be used in a job that has more than one job step.

PROCLIB
: specifies a private library of cataloged procedures to be searched before the system library of cataloged procedures is searched. See your on-site SAS support personnel for information about whether the PROCLIB ddname convention is used at your facility.

SORTLIB
: is used by some host sort utilities.

SORTMSG
: is used by some host sort utilities to print messages.

SORTWK*nn*
: specifies sort work data sets for the host sort utility. If allocated, this ddname is used instead of the SASSWK*nn* data sets.

STEPCAT
: specifies a private catalog that the operating environment is to use instead of the system catalog for the current job step.

SYSABEND
: in the event of an abnormal job termination, SYSABEND specifies a data set that receives a medium-sized dump that consists of user-allocated storage and modules, system storage related to current tasks and open files, and system and programs related to the terminated job. See also SYSMDUMP and SYSUDUMP below.

SYSHELP
: is used by TSO HELP libraries (not the SAS HELP facility).

SYSLIB
is used by some IBM system utility programs.

SYSMDUMP
in the event of an abnormal job termination, SYSMDUMP specifies a data set that receives a system dump in IPCS format. The contents of the dump are determined by z/OS installation options, although SYSMDUMP generally includes all user-allocated storage, all system-allocated storage used to control job execution, and all program modules (system modules and user programs) that were in use at the time the dump was taken.

SYSOUT
is used by some utility programs to identify an output data set.

SYSPRINT
is used by some utility programs to identify a data set for listings and messages that might be sent to the printer.

SYSUADS
is used by some TSO commands that might be invoked under SAS software.

SYSUDUMP
in the event of an abnormal job termination, SYSUDUMP specifies a data set that receives a "short" system dump that consists of user-allocated storage and modules and system storage related to current tasks and open files. See also SYSABEND and SYSMDUMP above.

SYS*nnnnn*
is reserved for internal use (for dynamic allocation) by the operating environment.

Using the SAS Remote Browser

What Is the Remote Browsing System?

The remote browsing system enables users who access SAS through a 3270 emulator (or a real 3270) to view SAS documentation from a Web browser on the user's PC. Previously, all documentation was displayed by the itemstore help in the SAS Help Browser window in the 3270 display. By displaying this documentation in your Web browser, you have better browsing capability and more complete documentation content.

Starting the Remote Browser Server

Remote browsing is invoked when SAS displays HTML output, usually from ODS, the Help system, or from the WBROWSE command. SAS attempts to detect your computer's network address and send remote browser requests to it. If you have not installed the remote browser server on your computer, SAS displays a dialog box that contains the address that is necessary to download the installer. The help server provides the remote browser installer, so you do not have to end your SAS session to install the remote browser server. Copy the address in the dialog box to your browser, download the installer, and run it. The installer places the remote browser server in the **`Startup Items`** folder, so that it will start each time that you start your computer.

Setting Up the Remote Browser

Overview of Setting Up the Remote Browser

After the remote browser server is running on your computer, you can run the help by using the defaults for the HELPBROWSER, HELPHOST, and HELPPORT system options. If the HELPHOST option is not coded, SAS attempts to connect to the remote browser at the network address to which your computer's 3270 emulator (or your 3270 terminal) points. If the address is not the correct address for the remote browser, you will have to set the appropriate value for the HELPHOST option.

- The HELPBROWSER system option specifies whether you want to use the new help (**REMOTE**, the default) or the traditional itemstore-based help (SAS) that uses the SAS Help browser. See the *SAS Language Reference: Dictionary* for more information.
- The HELPHOST system option specifies the network name of your computer, which runs the remote browser server. If the HELPHOST option value is not specified, it defaults to the network address of the computer that is running your 3270 display emulator. This computer is usually the same computer that is running the remote browser server. See "HELPHOST System Option" on page 541 for more information.
- The HELPPORT system option specifies the port number that the remote browser server is listening on. The default port is 3755, the port that is registered for the Remote Browser Server. See the *SAS Language Reference: Dictionary* for more information.

You can set these options at SAS invocation, in your configuration file, or during your SAS session in the OPTIONS statement or in the SAS System Options window.

Example 1: Setting Up the Remote Browser at SAS Invocation

The following code shows you how to set up the remote browser at SAS invocation:

```
sas o('helphost=mycomputer')
```

Example 2: Setting Up the Remote Browser during a SAS Session

The following code shows you how to set up the remote browser during your SAS session:

```
options helphost=mycomputer;
```

Remote Browsing and Firewalls

For General Users

If your network has a firewall between desktop computers and the computer that hosts SAS, browsers cannot display Web pages from your SAS session. Usually this is indicated by a timeout or connection error from the Web browser. If you receive a timeout or connection error, contact your system administrator.

For System Administrators

To enable the display of Web pages when a firewall exists between desktop computers and SAS, a firewall rule must be added that allows a browser to connect to SAS. The firewall rule specifies a range of network ports for which SAS remote browsing connections are allowed. Contact the appropriate administrator who can select and configure a range of firewall ports for remote browsing. The range size depends on the number of simultaneous SAS users. A value of approximately three times the number of simultaneous users should reserve a sufficient amount of network ports.

Once the firewall rule has been added, SAS must be configured to listen for network connections in the port range. Normally, SAS selects any free network port, but the HTTPSERVERPORTMIN and the HTTPSERVERPORTMAX system options limit the ports that SAS selects. Add these options to your SAS configuration file. Set the HTTPSERVERPORTMIN option to the lowest port in the range. Set the HTTPSERVERPORTMAX option to the highest port in the range. For example, if the network administrator defines a port range of 8000–8200, the system options are set as follows:

```
httpserverportmin=8000
httpserverportmax=8200
```

After the firewall rule is added and these system options are set, desktop computers can view Web pages through the firewall. If there are insufficient ports or the system options are specified incorrectly, a message displays in the SAS log.

For more information about these options, see HTTPSERVERPORTMIN= System Option and HTTPSERVERPORTMAX= System Option in the *SAS Language Reference: Dictionary*.

Converting Itemstore Help to HTML Help

Overview of Converting Itemstore Help

The SAS 9.2 remote browser does not read itemstore help files. The SAS %ISHCONV macro enables you to convert your itemstore help files into HTML files that you can use with the remote browser.

Note: If your location uses itemstore help files with SAS, then your SAS system programmer is the person who usually converts the itemstores to HTML files. Contact your system programmer if you cannot access the HTML help files. △

Creating a Common Directory

In order for SAS users at your location to access your HTML help files with the remote browser, you need to create a common directory in UFS. The common directory contains the HTML files that are created by the %ISHCONV macro. It can also contain subdirectories that contain more HTML files. The %ISHCONV macro uses the **htmdir** parameter to specify the common directory if you do not create it before running the macro. However, you have to specify a directory pathname for the **htmdir** parameter, so it is best to create the directory before you convert the itemstore files.

The common directory should allow all SAS users at your location to access the files, so you should place it in a directory path that is accessible to them. As always, it is a good idea to determine the best location for the directory before you create it and put your files in it. Such planning can prevent you from having to move the directory and its files at a later date.

Converting Your Files to HTML

The SAS macro, %ISHCONV, converts your itemstore files into HTML files. %ISHCONV uses the **ishelp** and **ishref** parameters to specify the data set name of the catalog that contains the itemstore help and the member of the itemstore help. It uses the **exphlp** and **htmdir** parameters to specify the filename of a work file for conversion processing and the pathname for the HTML files.

Note: The converted HTML files have a file extension of **.htm**. △

For more information about using %ISHCONV to convert your itemstores to HTML, see "%ISHCONV Macro" on page 342.

Adding HELPLOC Path Values

The INSERT system option enables you to add new path values to the HELPLOC option. Path values that are added with the INSERT option are read before paths that are already assigned to the HELPLOC option in your configuration file. You can insert multiple paths for the HELPLOC option.

The following commands insert a new path value for the HELPLOC option before other paths that are specified in your configuration file. The following example inserts a path in a directory that contains files in ASCII:

```
-insert (helploc='/u/userid/ishconv_dir_ascii')
```

The following example inserts a path in a directory that contains files in EBCDIC:

```
-insert (helploc='/u/userid/ishconv_dir_ed--1047;ebcdic')
```

After you use the INSERT command to assign additional path values, you can issue the following command:

```
proc options option=helploc; run;
```

To display the new value for HELPLOC that combines the two paths:

```
HELPLOC=( '/u/userid/ishconv_dir_ascii'
        '/u/userid/ishconv_dir_ed-1047;ebcdic'
        '/usr/local/SASdoc' )
```

The path value **'/usr/local/SASdoc'** is the value that was set for HELPLOC in your configuration file.

Accessing Your HTML Help Files

After your itemstore help files are converted to HTML, you can access the HTML help files by the same methods that you used to access the itemstore help:

- Select Help from the SAS menu bar.
- Enter the HELP command from the SAS command line.

The SAS HELP command:

```
help helploc://user.hlp/index.htm
```

accesses the **index.htm** file in the UFS directory that has the fully qualified name:

```
/u/userid/ishconv_dir_ascii/user.hlp/index.htm
```

Note: You are not limited to using the HELP command to access only the **index.htm** file of your help. You can issue the HELP command to access Help with the remote browser for Windows, SAS language elements, and so on, the same as you have with previous SAS Help systems. △

If a help file with the same relative path and filename exists in multiple HELPLOC path values, SAS displays the file from the first path that is encountered. This feature enables you to amend an existing file that is available to SAS.

If you get an error message that the help is not available, use the HELPLOC system option to specify the location of the help files. You can include the HELPLOC option in your configuration file, or you can issue it at SAS invocation. Contact your system programmer for the location of the help files that you need to use with the HELPLOC option.

See Also

- "%ISHCONV Macro" on page 342
- Chapter 15, "Macros under z/OS," on page 335
- HELPBROWSER System Option in *SAS Language Reference: Dictionary*
- "HELPLOC= System Option" on page 542
- "INSERT= System Option" on page 546
- *SAS Macro Language: Reference*

Creating User-Defined Help Files in HTML

You can write your own HTML help files to use with SAS. Use the same tools or file editors to write these HTML files that you would use to write any other HTML files. After you have written these files, place them in a location where SAS can access them. If your HTML help files are encoded in EBCDIC, you need to include the **`;ebcdic`** attribute in the declaration for the HELPLOC system option. For information about working with ASCII-encoded files in the z/OS USS environment, see IBM's *z/OS UNIX System Services User's Guide*.

For information about using the HELPLOC system option, see "HELPLOC= System Option" on page 542.

Note: SAS provides the ITEMS procedure to enable you to produce itemstore files, but we encourage you to create and use HTML help files. Any itemstore file that you create is used with SAS itemstore files that have not been updated for this release of SAS. △

Using Remote Browsing with ODS Output

The SAS Output Delivery System can be used to generate graphical reports of your SAS data. Remote browsing enables you to view your output directly from the SAS session, either in real time as the output is generated, or on demand from the Results window.

Remote browsing displays ODS output that is generated to z/OS native data sets (sequential, PDS, or PDSE) or a UFS directory. HTML, PDF, RTF, and XLS file types are displayed with the remote browsing system. If your browser does not have the appropriate plug-in for non-HTML data types, it will display a download dialog box rather than the actual data. This dialog box will allow you to download the report to your PC and view it using a local program, such as Excel for an XLS file.

Note: When images or graphics are written to a z/OS native data set and remote browsing is being used to view the output, the URL=NONE option should not be used with the ODS statement. Using this option causes the HTML to be written with incomplete filenames, and the remote browsing system is not able to determine the location of the image or graphics. When this situation occurs, the browser displays broken image icons in the HTML output. △

The automatic display of ODS output is turned off by default. You can turn on the automatic display of ODS output by issuing the AUTONAVIGATE command in the Results window.

For more information about viewing ODS output with a browser, see "Viewing ODS Output on an External Browser" on page 135.

Using Itemstore Help Files

Accessing SAS Itemstore Help Files

Note: SAS supports itemstore help files for SAS 9.2. However, support for itemstore help files will be removed in a future release of SAS. Updated help files for SAS 9.2 are available only in HTML. If you use your itemstore help files for SAS 9.2, please remember that the itemstore help files that are provided by SAS have not been updated. We strongly recommend that you convert your itemstore help files to HTML when you install SAS 9.2, and use your HTML help files with the updated SAS help files that are in HTML. See "Converting Itemstore Help to HTML Help" on page 35 and "%ISHCONV Macro" on page 342 for information about converting your itemstore files to HTML. △

Help is available through the SAS online Help facility. To obtain host-specific help, execute the PMENU command as necessary to display SAS menus, then select **Help ▸ SAS System Help ▸ Main TOC ▸ Using SAS Software in z/OS**. Then select topics of interest at increasing levels of detail.

Issue the KEYS command to determine the function keys used to page up, down, left, and right through help pages, and to move backward and forward between Help topics.

Using User-Defined Itemstore Help Files

Your site might provide user-defined help that provides site-specific information via the standard SAS help browser. To access user-defined help via the SAS help browser, you need to allocate a user-defined help library at SAS invocation.

The user-defined help library contains help information in the form of one or more *itemstores*, which use a file format that allows SAS to treat the itemstore as a file system within a file. Each itemstore can contain directories, subdirectories, and individual Help topics. For information about loading user-defined help into itemstores, refer to "ITEMS Procedure" on page 368.

Help for SAS software is contained in itemstores. SAS automatically allocates libraries for SAS software help at SAS invocation. To invoke SAS so that it recognizes user-defined help, follow these steps:

1. In an autoexec file, allocate the SAS library that contains the user-defined itemstores using the LIBNAME statement. For example, if the libref is to be MYHELP and the itemstore is named APPL.HELP.DATA, the LIBNAME statement in the SAS invocation would be

   ```
   libname myhelp 'appl.help.data' disp=shr;
   ```

   See "Autoexec Files" on page 11 and "LIBNAME Statement" on page 451 for details.

2. Concatenate your itemstores to the SAS help itemstore named by the HELPLOC= system option at SAS invocation. For example, if the libref for your user-defined

help was MYHELP, and if the itemstore in the libref was named PRGAHELP, then the HELPLOC= specification in the SAS invocation would be as follows:

```
helploc='myhelp.prgahelp'
```

See "HELPLOC= System Option" on page 542 for details on the HELPLOC= system option.

User-defined help cannot be added to the SAS help itemstore because most users have read-only access to the SAS help library.

After SAS has been invoked so that it can recognize user-defined help, you can access that help with the standard SAS help browser by issuing the HELP command and specifying the appropriate universal resource locator (URL). For example, if the Help topic that you want to display is named DIRAHLP1.HTM, and if that Help topic is contained in an itemstore directory named PRGADIRA, the HELP command would be as follows:

```
help helploc://prgadira/dirahlp1.htm
```

See the next section for information about developing user-defined help for the SAS help browser.

Creating User-Defined Itemstore Help Files

You can create help for your site or for your SAS programs that can be displayed in the standard SAS help browser. To ensure that your user-defined help will be displayed as it is written, use only the subset of tags from HTML that are supported on the SAS help browser. Help information in tags that are not supported by the SAS help browser might be ignored by the SAS help browser.

The following table describes the HTML tags supported by the SAS help browser. The TABLE tag is the only frequently used tag that is not supported at this time. To add tables to your help, use the PRE tag and format the text manually using blank spaces, vertical bars, dashes, and underscores as needed.

Table 1.5 HTML Tags Supported by the SAS Help Browser

Tag Type	Tag Names	Description
heading	H1, H2, H3, H4, H5, H6	for hierarchical section headings
paragraph	P	for text in the body of a help file
list	UL, OL, DIR, MENU	for unordered (bullet) lists, ordered (numbered) lists, directory (unordered, no bullets) lists, and menu (unordered) lists
definition list	DL, DT, DD	for definition lists, titles of items, and definitions of items
preformatted text	PRE, XMP, LISTING	for tables, which must be manually formatted with blank spaces
font specification	I, B, U	for italic, bold, and underlined text
phrase	EM, STRONG, DFN, CODE, SAMP, KBD, VAR, CITE	for emphasis, strong emphasis, definitions, code examples, code samples, keyboard key names, variables, citations

Tag Type	Tag Names	Description
link	A, LINK	for anchors and the links that reference those anchors
document	TITLE, BASE, HEAD, HTML	for titles in the browser, base URLs, heading sections at the top of a page

For information about the options available for these tags, see any reference for the version of HTML supported by your browser.

For information about loading your help into itemstores, see "ITEMS Procedure" on page 368.

Exiting or Terminating Your SAS Session in the z/OS Environment

Preferred Methods for Exiting SAS

These are the preferred methods for exiting a SAS session:

- select **File ▸ Exit**
- use **`endsas;`**
- enter **`BYE`** in the command line.

Additional Methods for Terminating SAS

In addition to the preferred methods for exiting a SAS session, when SAS is running on a server a system operator can terminate it in the following ways:

STOP
: This method is the equivalent of an application requesting a normal shutdown. You should have no problems with your files.

CANCEL
: The operating system initiates the termination of SAS, but application error handlers can still run and cleanup is possible. Your files will be closed, and the buffers will be flushed to disk. However, there is no way to ensure that the shutdown will always be orderly. Your files could be corrupted.

MEMTERM (also known as FORCE)
: The operating system terminates all application processes with no recovery. This is the equivalent to what would happen if the system were rebooted.

Some databases, such as DB2, are able to recover from both the CANCEL and MEMTERM types of failures. These applications accomplish this task by logging every change so that, regardless of when a failure occurs, the log can be replayed to enable recovery to a valid state. However, some transactions could still be lost.

Although you can terminate SAS using these techniques, you should try one of the three preferred techniques listed first.

See Also

"What If SAS Doesn't Start?" on page 7.

Solving Problems under z/OS

Overview of Solving Problems under z/OS

As you use SAS software under z/OS, you might encounter many different types of problems. Problems might occur within your SAS program, or they might be with some component of the operating environment or with computer resources rather than with SAS software. For example, problems might be related to job control language or to a TSO command.

Problems Associated with the z/OS Operating Environment

If a problem is detected by the operating environment, it sends messages to the job log or to the terminal screen (not the SAS log). In this case, you might need to consult an appropriate IBM manual or your on-site systems staff to determine the problem and the solution.

Most error messages indicate which part of the operating environment is detecting the problem. Here are some of the most common message groups, along with the operating environment component or utility that issues them:

ARC
: IBM Hierarchical Storage Manager

BPX
: IBM UNIX System Services

CEE
: IBM Language Environment (LE)

CSV*xxxx*
: z/OS load module management routines

FSUM
: IBM UNIX System Services Shell and Utilities

ICE*xxxxx*
: IBM sort utility

ICH*xxxx*
: RACF system-security component of z/OS

IDC*xxxxx*
: catalog-management component of z/OS

IEC*xxxxx*
: z/OS data-management routines

IEF
: IBM JCL Interpreter

IKJ*xxxx*
: TSO terminal monitor program (TMP) and other TSO components

LSC
: SAS/C Runtime Library

WER*xxxxx*
: SYNCSORT program

Consult the appropriate system manual to determine the source of the problem.

Overview of Solving Problems within SAS Software

Several resources are available to help you if you determine that your problem is within SAS software. These resources are discussed in the following sections.

Examining the SAS Log

The primary source of information for solving problems that occur within SAS software is the SAS log. The log lists the SAS source statements along with notes about each step, warning messages, and error messages. Errors are flagged in the code, and a numbered error message is printed in the log. It is often easy to find the incorrect step or statement just by glancing at the SAS log.

Note: Some errors require that diagnostic messages are written before the SAS log is opened or after it is closed. Such messages are written to the SASCLOG data set. Under TSO, SASCLOG is normally allocated to the terminal. Occasionally, operating system error messages might be issued during execution of a SAS program. These messages appear in the job log or, under TSO, on the terminal. △

Checking the Condition Code

Upon exit, SAS returns a condition code to the operating environment that indicates its completion status. The condition code is translated to a return code that is meaningful to the operating environment. SAS issues the condition codes in the following table:

Table 1.6 z/OS Condition Codes

Return Code	Meaning
0	Successful completion
4	WARNING messages issued
8	Nonfatal ERROR messages issued
12	Fatal ERROR messages issued
16	ABORT; executed
20	ABORT RETURN; executed
ABND	ABORT ABEND; executed

DATA Step Debugger

The DATA step debugger is an interactive tool that helps you find logic errors, and sometimes data errors, in SAS DATA steps. By issuing commands, you can execute DATA step statements one by one or in groups, pausing at any point to display the resulting variable values in a window. You can also bypass the execution of one or more statements. For further information about the DATA step debugger, see the *SAS Language Reference: Dictionary*.

Using SAS Statements, Procedures, and System Options to Identify Problems

If you are having a problem with the logic of your program, there might be no error messages or warning messages to help you. You might not get the results or output that you expect. Using PUT statements to write messages to the SAS log or to dump the values of all or some of your variables might help. Using PUT statements enables you to follow the flow of the problem and to see what is going on at strategic places in your program.

Some problems might be data related; these can be difficult to trace. Notes that appear in the SAS log following the step that reads and manipulates the data might be very helpful. These notes provide information such as the number of variables and observations that were created. You can also use the CONTENTS and PRINT procedures to look at the data definitions as SAS recorded them or to look at all or parts of the data in question.

SAS system options can also assist with problem resolution. Refer to the *SAS Language Reference: Dictionary* for details on the following system options and others that affect problem resolution:

MLOGIC
controls whether SAS traces execution of the macro language processor.

MPRINT
displays SAS statements that are generated by macro execution.

SOURCE
controls whether SAS writes source statements to the SAS log.

SOURCE2
writes secondary source statements from included files to the SAS log.

SYMBOLGEN
controls whether the results of resolving macro variable references are written to the SAS log.

Host-System Subgroup Error Messages

See “Messages from the SASCP Command Processor” on page 667 for brief explanations of many of the host-system subgroup error messages that you might encounter during a SAS session.

The z/OS system log can also contain useful information that might assist you with diagnosing a problem with SAS. Consult your system administrator for assistance with viewing the system log.

Support for SAS Software

Overview of Support for SAS Software

Support for SAS software is shared by SAS and your installation or site. SAS provides maintenance for the software; the SAS Installation Representative, the on-site SAS support personnel, and the SAS Training Coordinator for your site are responsible for giving you direct user support.

- The SAS Installation Representative receives all shipments and correspondence and distributes them to the appropriate personnel at your site.

- The on-site SAS support personnel are knowledgeable SAS users who support the other SAS users at your site. The SAS Technical Support Division is available to assist your on-site SAS support personnel with problems that you encounter.
- The SAS Training Coordinator works with the SAS Education Division to arrange training classes for SAS users.

Working with Your On-Site SAS Support Personnel

At your site, one or more on-site SAS support personnel have been designated as the first point of contact for SAS users who need help resolving problems.

If the on-site SAS support personnel are unable to resolve your problem, then the on-site SAS support personnel contact the SAS Technical Support Division for you. In order to provide the most efficient service possible, the company asks that you do not contact SAS Technical Support directly.

SAS Technical Support

The SAS Technical Support Division can assist with suspected internal errors in SAS software and with possible system incompatibilities. It can also help answer questions about SAS statement syntax, general logic problems, and procedures and their output. However, the SAS Technical Support Division cannot assist with special-interest applications, with writing user programs, or with teaching new users. It is also unable to provide support for general statistical methodology or for the design of experiments.

Generating a System Dump for SAS Technical Support

Follow these steps to generate a system dump that can be interpreted by SAS Technical Support:

1. Disable ABEND-AID or any other dump formatting system software before generating the dump.
2. Create a sequential data set with the DCB attributes DSORG=PS RECFM=FB LRECL=256 and the following contents:

```
reset
set tkopt_dumpprol=
set tkopt_nostae=
set tkopt_nostaex=
```

3. In the batch job or TSO session in which SAS is started, allocate the following ddname's:
   - Allocate the ddname TKMVSENV to the sequential data set that is described above.
   - If an unformatted dump is desired, which is normally the case unless otherwise advised by SAS Technical Support, allocate the ddname SYSMDUMP to a disk data set. Specifying **`SPACE=(CYL,(50,50))`** is usually sufficient. In batch, it is usually convenient to allocate the dump data set **`DISP=(,DELETE,CATLG)`** so that it will be created only if the job abends.
   - If a formatted dump is desired or requested, instead of an unformatted dump, allocate the ddname SYSUDUMP to a disk data set or an appropriate SYSOUT class. In most cases, this would be a SYSOUT class that is not automatically printed.

- Specify the following options at SAS invocation: NOSTAE, DUMPPROL, SOURCE, SOURCE2, NOTES, MPRINT, and SYMBOLGEN.

To deliver the dump to SAS, use one of the following methods:

FTP
: Send unformatted dumps in BINARY mode and inform SAS Technical Support of the DCB attributes of the original dump data set. Send formatted dumps in ASCII mode.

Tape
: Use IEBGENER to copy the dump data set to a magnetic tape cartridge using IBM standard labels.