CHAPTER

16 Statements

Statements Under UNIX

This chapter describes SAS statements that exhibit behavior or syntax that is specific to UNIX environments. Each statement description includes a brief "UNIX specifics" section that tells which aspect of the statement is specific to UNIX. Each statement is described in both this documentation and in *SAS Language Reference: Dictionary*.

ABORT

Stops executing the current DATA step, SAS job, or SAS session

Valid: in a DATA step

UNIX specifics: values of *n*

Syntax

ABORT <ABEND|RETURN><*n*>;

Details

The *n* option allows you to specify the value of the exit status code that the SAS System returns to the shell when it stops executing. The value of *n* can range from 0 to 255.

See Also

- "Determining the Completion Status of a SAS Job" on page 18

ATTRIB

Associates a format, informat, label, and/or length with one or more variables

Valid: in a DATA step

UNIX specifics: length specification

Syntax

ATTRIB *variable-list*-1 *attribute-list*-1... <*variable-list-n attribute-list-n*>;

Details

The minimum length that you can specify for a numeric variable depends on the floating-point format used by your system. Because most systems use the IEEE floating-point format, the minimum is 3 bytes.

See Also

- □ Chapter 8, "Data Representation," on page 159

FILE

Specifies the current output file for PUT statements

Valid: in a DATA step

UNIX specifics: *file-specification* and *host-options*

Syntax

FILE *file-specification* <*options*> <*host-options*>;

file-specification
can be any of the file specification forms discussed in "Accessing an External File or Device" on page 104 .

options
can be any of the portable options for the FILE statement. See *SAS Language Reference: Dictionary* for a description of these options.

host-options
are specific to UNIX environments. These options can be any of the following:

BLKSIZE=
BLK=
specifies the number of bytes that are physically written in one I/O operation. The default is 8K. The maximum is 1G-1.

LRECL=
: specifies the logical record length. Its meaning depends on the record format in effect (RECFM). The default is 256. The maximum length is 1G.

- □ If RECFM=F, the value for the LRECL= option determines the length of each output record. The output record is truncated or padded with blanks to fit the specified size.
- □ If RECFM=N, the value for the LRECL= option must be at least 256.
- □ If RECFM=V, the value for the LRECL= option determines the maximum record length. Records that are longer than the specified length are divided into multiple records.

MOD
: indicates that data written to the file should be appended to the file.

NEW|OLD
: indicates that a new file is to be opened for output. If the file already exists, it is deleted and re-created. This is the default action.

RECFM=
: specifies the record format. Values for the RECFM= option are

D	default format (same as variable).
F	fixed format. That is, each record has the same length. Do not use RECFM=F for external files that contain carriage-control characters.
N	binary format. The file consists of a stream of bytes with no record boundaries.
P	print format. The SAS System writes carriage–control characters.
V	variable format. Each record ends in a newline character.
S370V	variable S370 record format (V).
S370VB	variable block S370 record format (VB).
S370VBS	variable block with spanned records S370 record format (VBS).

UNBUF
: tells the SAS System not to perform buffered writes to the file on any subsequent FILE statement. This option applies especially when writing to a data collection device.

See Also

- □ Chapter 5, "Using External Files and Devices," on page 103

FILENAME

Associates a SAS fileref with a device or an external file

Valid: anywhere

UNIX specifics: *device-type*, *external-file*, and *host-options*

Syntax

FILENAME *fileref* <*device-type* > '*external-file*' <*host-options*>;

FILENAME *fileref device-type* <'*external-file*'> <'*host-options*'>;

FILENAME *fileref* CLEAR | _ALL_ CLEAR;

FILENAME *fileref* LIST | _ALL_ LIST;

FILENAME *fileref* ('*pathname-1*' ... '*pathname-n*');

device-type
specifies a device for the output. It can be any one of the devices listed in Table 16.1 on page 239. DISK is the default device type. If you are associating the fileref with a DISK file, you do not need to specify the device type.

'external-file'
differs according to device type. Table 16.1 on page 239 shows the information appropriate to each device. Remember that UNIX filenames are case-sensitive.

'pathname-1'...'pathname-n'
are pathnames for the files that you want to access with the same fileref. Use this form of the FILENAME statement when you want to concatenate filenames. Concatenating filenames is available only for DISK files, so you do not have to specify the *device-type*. Separate the pathnames with either commas or blank spaces.

'host-options'
are options specific to UNIX. They can be any of the following:

BLKSIZE=
BLK=
specifies the number of bytes that are physically written or read in one I/O operation. The default is 8K. The maximum is 1G-1. If you specify RECFM=S370VBS, you should specify BLKSIZE=32760 to avoid errors with records longer than 255.

LRECL=
specifies the logical record length. Its meaning depends on the record format in effect (RECFM). The default is 256. The maximum length is 1G.

- If RECFM=F, the value for the LRECL= option determines either the number of bytes to be read as one record or the length of each output record. The output record is truncated or padded with blanks to fit the specified size.
- If RECFM=N, the value for the LRECL= option must be at least 256.
- If RECFM=V, the value for the LRECL= option determines the maximum record length. Records that are longer than the specified length are divided into multiple records on output and truncated on input.
- If RECFM=S370VBS, you should specify LRECL=32760 to avoid errors with records longer than 255.

MOD
indicates that data written to the file should be appended to the file.

NEW|OLD
indicates that a new file is to be opened for output. If the file already exists, it is deleted and re-created. This is the default action.

RECFM=
specifies the record format. Values for the RECFM= option are

D	default format (same as variable).
F	fixed format. That is, each record has the same length. Do not use RECFM=F for external files that contain carriage-control characters.
N	binary format. The file consists of a stream of bytes with no record boundaries.
P	print format. On output, the SAS System writes carriage-control characters.
V	variable format. Each record ends in a newline character.
S370V	variable S370 record format (V).
S370VB	variable block S370 record format (VB).
S370VBS	variable block with spanned records S370 record format (VBS). If you specify RECFM=S3270VBS, then you should specify BLDSIZE=32760 and LRECL=32760 to avoid errors with records longer than 255.

Note: To use the S370V, S370VB, or S370VBS format to access a file that was created under the OS/390 operating environment, the file must be of type RECFM=U. △

This option is used for both input and output.

UNBUF
tells the SAS System not to perform buffered writes to the file on any subsequent FILE statement. This option applies especially when reading from or writing to a data collection device. As explained in *SAS Language Reference: Dictionary*, it also prevents buffered reads on INFILE statements.

Table 16.1 Device Information in the FILENAME Statement

Device or Access Method	Function	External-file
CATALOG	references a SAS catalog as an external file.	is a valid two-, three-, or four-part SAS catalog name followed catalog options (if needed). See *SAS Language Reference: Dictionary* for details.
DISK	associates the fileref with a DISK file.	is either the pathname for a single file or, if you are concatenating filenames, a list of pathnames separated by blanks or commas and enclosed in parentheses. The level of specification depends on your location in the file system. Table 4.2 on page 88 shows character substitutions you can use when specifying a UNIX pathname.
DUMMY	associates a fileref with a null device.	none. DUMMY allows you to debug your application without reading or writing to a device. Output to this device is discarded.
EMAIL	sends electronic mail to an address.	is an address and e-mail options. See “Sending Electronic Mail from Within the SAS System (EMAIL)” on page 116 for more information.

Device or Access Method	Function	External-file
FTP	reads or writes to a file from any machine on a network that is running an FTP server.	is the pathname of the external file on the remote machine followed by FTP options. See *SAS Language Reference: Dictionary* and “Assigning Filerefs to Files on Other Systems (FTP and SOCKET access types)” on page 111 for details. If you are transferring a file from the OS/390 operating environment and you want to access that file using any of the S370 formats, then the file must be of type RECFM=U before you transfer it to UNIX.
PIPE	reads input from or writes output to a UNIX command.	is a UNIX command. See Chapter 6, “Routing Output,” on page 125 for details.
PLOTTER	sends output to a plotter.	is a device name and plotter options. See “Using PRTFILE and PRINT with a Fileref” on page 131and “Using the PRINTTO Procedure” on page 133 for details.
PRINTER	sends output to a printer.	is a device name and printer options. See “Using PRTFILE and PRINT with a Fileref” on page 131and “Using the PRINTTO Procedure” on page 133 for details.
SOCKET	reads and writes information over a TCP/IP socket.	depends on whether the SAS application is a server application or a client application. In a client application, *external-file* is the name or IP address of the host and the TCP/IP port number to connect to followed by any TCP/IP options. In a server application, it is the port number to create for listening, followed by the SERVER keyword, and then any TCP/IP options. See *SAS Language Reference: Dictionary* for details.
TAPE	associates a fileref with a tape.	is the pathname for a tape device. The name specified should be the name of the special file associated with the tape device. See “Processing Files on TAPE” on page 121 for more information.
TEMP	associates a fileref with an external file stored in the WORK data library.	none.
TERMINAL	associates a fileref with a terminal.	is the pathname of a terminal.

URL	allows you to access remote files using the URL of the file.	is the name of the file that you want to read from or write to on a URL server. The URL must be in one of these forms: **`http://`*`hostname/file`*** **`http://`*`hostname:portno/file`***
XPRINTER	sends output to the default printer that was set up through the Printer Setup dialog box.	none.

See Also

- Chapter 5, "Using External Files and Devices," on page 103
- Chapter 6, "Routing Output," on page 125

FOOTNOTE

Prints up to ten lines of text at the bottom of the procedure output

Valid: anywhere

UNIX specifics: maximum length of footnote

Syntax

FOOTNOTE <*n*> <'*text*' | "*text*">;

Details

The maximum footnote length is 255 characters. If the length of the specified footnote is greater than the value of the LINESIZE option, the SAS System truncates the footnote to the line size.

%INCLUDE

Includes and executes SAS statements and data lines

Valid: anywhere

UNIX specifics: *source*, if a file specification is used

Syntax

%INCLUDE *source-1* <...*source-n*> </<SOURCE2> <S2=*length*> <*host-options*>>;

source

describes the location you want to access with the %INCLUDE statement. The three possible sources are a file specification, internal lines, or keyboard entry. The file specification can be any of the file specification forms that are discussed in “Accessing an External File or Device” on page 104 .

host-options
There are no options specific to UNIX for this statement.

See Also

- Chapter 5, “Using External Files and Devices,” on page 103

INFILE

Specifies an external file to be read with an INPUT statement

Valid: in a DATA step

UNIX specifics: *file-specification* and *host-options*

Syntax

INFILE *file-specification* <*options*> <*host-options*>;

INFILE *DBMS-specification*;

file-specification
can be any of the file specification forms discussed in “Accessing an External File or Device” on page 104 .

host-options
are options specific to UNIX. They can be any of the following:

BLKSIZE=
BLK=
specifies the number of bytes that are physically read in one I/O operation. The default is 8K. The maximum is 1G-1.

LRECL=
specifies the logical record length. Its meaning depends on the record format in effect (RECFM). The default is 256. The maximum length is 1G.

- If RECFM=F, the value for the LRECL= option determines the number of bytes to be read as one record.
- If RECFM=N, the value for the LRECL= option must be at least 256.
- If RECFM=V, the value for the LRECL= option determines the maximum record length. Records that are longer than the specified length are truncated on input.

RECFM=
specifies the record format. Values for the RECFM= option are

D	default format (same as variable).
F	fixed format. That is, each record has the same length.
N	binary format. The file consists of a stream of bytes with no record boundaries.

P	print format.
V	variable format. Each record ends in a newline character.

See Also

□ Chapter 5, “Using External Files and Devices,” on page 103

LENGTH

Specifies the number of bytes that the SAS System uses to store a variable’s value

Valid: in a DATA step

UNIX specifics: valid numeric variable lengths

Syntax

LENGTH <*variable-1*><*...variable-n*> <$> *length* <DEFAULT=*n*>

length
can range from 3 to 8 for numeric variables under UNIX. The minimum length you can specify for a numeric variable depends on the floating-point format used by your system. Because most systems use the IEEE floating-point format, the minimum is 3 bytes.

DEFAULT=*n*
changes the default number of bytes that are used for storing the values of newly created numeric variables from 8 to the value of *n*. Under UNIX, *n* can range from 3 to 8.

See Also

□ Chapter 8, “Data Representation,” on page 159

LIBNAME

Associates or disassociates one or more SAS data libraries with a libref; lists the characteristics of a SAS data library

Valid: anywhere

UNIX specifics: *engine*, *library*, and *engine/host-options*

Syntax

LIBNAME *libref* <*engine*> ’*SAS-data-library*’ <*options*> <*engine/host-options*>;

LIBNAME *libref* <*engine*> (’*library-1*’<,...*library-n*>) <*options*>;

LIBNAME *libref* (*'library-1'* | *libref-1,...,'library-n'* | *librefn*);

LIBNAME *libref* CLEAR | _ALL _ CLEAR;

LIBNAME *libref* LIST | _ALL _ LIST;

libref
is any valid libref as documented in *SAS Language Reference: Dictionary*. The SAS System reserves some librefs for special system libraries. See "Librefs Assigned by SAS" on page 94 for more information.

engine
is one of the library engines supported under UNIX. See "Details" on page 245 for a description of the engines. If no engine name is specified, the SAS System determines which engine to useas described in "Omitting Engine Names From the LIBNAME Statement" on page 91.

'SAS-data-library'
differs according to the engine that you specify and according to your current working directory. Table 16.2 on page 245 describes what each engine expects for this argument. Specify directory pathnames as described in "Specifying Pathnames" on page 87. You cannot create directories with the LIBNAME statement. The directory that you specify here must already exist, and you must have permissions to it. Enclose the data library name in quotes. Remember that UNIX pathnames are case-sensitive.

'library-n' | libref-n
are pathnames or librefs (that have already been assigned) for the data libraries that you want to access with one libref. Use these forms of the LIBNAME statement when you want to concatenate data libraries. Separate the pathnames with either commas or blank spaces. Enclose library pathnames in quotation marks. Do not enclose librefs in quotation marks. See "Assigning a Libref to Several Directories (Concatenating Directories)" on page 91 for more information .

options
are LIBNAME statement options that are available in all operating environments. See *SAS Language Reference: Dictionary* for information about these options.

engine/host-options
can be any of the options described in "Engine/Host Options" on page 246.

ALL
refers to all librefs currently defined. You can use this keyword when you are listing or clearing librefs.

CLEAR
clears the specified libref or, if you specify _ALL_, clears all librefs that are currently defined. SASUSER, SASHELP, and WORK remain assigned.

Note: When you clear a libref defined by an environment variable, the variable remains defined, but it is no longer considered a libref. You can still reuse it, either as a libref or a fileref. See "Using Environment Variables as Librefs" on page 93 for more information. △

The SAS System automatically clears the association between librefs and their respective data libraries at the end of your job or session. If you want to associate an existing libref with a different SAS data library during the current session, you do not have to end the session or clear the libref. The SAS System automatically reassigns the libref when you issue a LIBNAME statement for the new SAS data library.

LIST
prints to the SAS log the engine, pathname, file format, access permissions, and so on, that are associated with the specified libref or, if you specify _ALL_, prints this information for all librefs that are currently defined. Librefs defined as environment variables appear only if you have already used those librefs in a SAS statement.

Details

There are two main types of engines:

View engines
enable the SAS System to read SAS data views that are described by SAS/ACCESS software, the SQL procedure, and DATA step views. The use of SAS view engines is automatic because the name of the view engine is stored as part of the descriptor portion of the SAS data set.

Library engines
control access at the SAS data library level. Every SAS data library has an associated library engine, and the files in that library can be accessed only through that engine. There are two types of library engines:

native engines
access SAS files created and maintained by the SAS System. See Table 16.2 on page 245 for a description of these engines.

interface engines
treat other vendors' files as if they were SAS files. See Table 16.2 on page 245 and “Accessing BMDP, OSIRIS, or SPSS Data Files” on page 98 for more information.

Table 16.2 Engine Names and Descriptions

Engine Type	Name (Alias)	Description	SAS-data-library
default	V8 (BASE) V7	enables you to create new SAS data files and access existing SAS data files that were created with Version 7 or Version 8. The V7 and V8 engines are identical. This engine enables read access to data files that were created with some earlier versions of SAS, but this engine is the only one that supports Version 8 catalogs. This engine allows for data set indexing and compression and is also documented in *SAS Language Reference: Dictionary*.	is the pathname of the directory containing the library.
sequential	V8TAPE (TAPE) V7TAPE V6TAPE	accesses SAS data files that were created in a sequential format, whether on tape or on disk. This engine requires less overhead than the default engine because sequential access is simpler than random access. This engine is also documented in *SAS Language Reference: Dictionary*.	is the name of the special file (see “Introduction to External Files and Devices” on page 103) associated with the sequential device, such as **/dev/rmt/0mn**.
compatibility	V6	accesses any data file that was created by Release 6.07 through 6.12.	is the pathname of the directory containing the library.

Engine Type	Name (Alias)	Description	SAS-data-library
servers	SPDS	enables communication between a client session and a data server. You must have the Scalable Performance Data Server licensed on your client machine to use this engine. Refer to *Scalable Performance Data Server User's Guide, Version 2* for more information.	is the logical LIBNAME domain name for an SPDS data library on the server machine. The name server resolves the domain name into the physical path for the library.
	MDDB	enables communication between a client session and an MDDB server. You must have SAS/MDDB Server licensed either or your client machine or on your server machine to use this engine. Refer to *SAS MDDB Server Software: Administration Guide* for complete information.	
transport	XPORT	accesses transport data sets. This engine creates machine-independent SAS transport files that can be used under all hosts running Release 6.06 or later of the SAS System. This engine is documented in *Moving and Accessing SAS Files across Operating Environments*.	is the pathname of either a sequential device or a disk file.
interface	BMDP	provides read-only access to BMDP files. This engine is available only on AIX, HP-UX, and Solaris.	is the pathname of the data file.
	OSIRIS	provides read-only access to OSIRIS files.	is the pathname of the data file.
	SPSS	provides read-only access to SPSS files	is the pathname of the data file.

Engine/Host Options The LIBNAME statement accepts the FILELOCKS option:

FILELOCKS=NONE | FAIL | CONTINUE

This option specifies whether file locking is on or off for the library that you are defining. This LIBNAME statement option works like the FILELOCKS system option, except that it applies only to the library that you are defining. See “FILELOCKS” on page 266 for more information.

You can also specify any of the options supported by the SPDS engines. SPDS is the Scalable Performance Data Server. Refer to *Scalable Performance Data Server User's Guide, Version 2* for a description of these options.

See Also

- Chapter 4, “Using SAS Files,” on page 83

SYSTASK

Executes, lists, or kills asynchronous tasks

Alias: LISTTASK is an alias for SYSTASK LIST

Valid: anywhere

UNIX specifics: all

Syntax

SYSTASK COMMAND "*host-command*"
<WAIT|NOWAIT>
<TASKNAME=*taskname*>
<MNAME=*name-var*>
<STATUS=*stat-var*>
<SHELL<="*shell-command*">>
<CLEANUP>;

SYSTASK LIST <_ALL_ | *taskname*> <STATE> <STATVAR>;

SYSTASK KILL *taskname* <*taskname...*>;

COMMAND
executes the *host-command*.

LIST
lists either a specific active task or all of the active tasks in the system. A task is *active* if it is running or if it has completed and has not been waited for using the WAITFOR statement.

KILL
forces the termination of the specified task(s).

host-command
specifies the name of a UNIX command (including any command-specific options) or the name of an X Windows or Motif application. Enclose the command in either single or double quotes. If the command options require quotes, repeat the quotes. For example:

```
SYSTASK COMMAND "xdialog -m ""There was an error."" -t ""Error"" -o";
```

Note: The *host-command* that you specify cannot require input from the keyboard. △

WAIT | NOWAIT
determines whether SYSTASK COMMAND suspends execution of the current SAS session until the task has completed. NOWAIT is the default. For tasks that are started with the NOWAIT option, you can use the WAITFOR statement when necessary to suspend execution of the SAS session until the task has finished. See "WAITFOR" on page 250.

TASKNAME=*taskname*
specifies a name that identifies the task. Task names must be unique among all active tasks. A task is *active* if it is running or if it has completed and has not been waited for using the WAITFOR statement. Duplicate task names generate an error in the SAS log. If you do not specify a task name, SYSTASK will automatically generate a name. If the task name contains a blank character, enclose the task name in quotes.

Task names cannot be reused, even if the task has completed, unless you either issue the WAITFOR statement for the task or you specify the CLEANUP option.

MNAME=*name-var*
specifies a macro variable in which you want SYSTASK to store the task name that it automatically generated for the task. If you specify both the TASKNAME option

and the MNAME option, SYSTASK copies the name that you specified with TASKNAME into the variable that you specified with MNAME.

STATUS=*stat-var*
specifies a macro variable in which you want SYSTASK to store the status of the task. Status variable names must be unique among all active tasks.

SHELL<=“*shell-command*”>
specifies that the *host-command* should be executed with the host shell command. If you specify a *shell-command*, SYSTASK uses the shell command that you specify to invoke the shell; otherwise, SYSTASK uses the default shell. Enclose the shell command in quotes.

Note: The SHELL option assumes that the shell command that you specify uses the **-i** option to pass statements. Usually, your shell command will be **sh**, **csh**, **ksh**, or **bash**. △

CLEANUP
specifies that the task should be removed from the SYSTASK LIST output when the task completes. You can then reuse the task name without issuing the WAITFOR statement.

ALL
specifies all active tasks in the system.

STATE
displays the status of the task, which can be Start Failed, Running, or Complete.

STATVAR
displays the status variable associated with the task. The status variable is the variable that you assigned with the STATUS option in the SYSTASK COMMAND statement.

Details

SYSTASK allows you to execute host-specific commands from within your SAS session or application. Unlike the X statement, SYSTASK runs these commands as *asynchronous* tasks, which means that these tasks execute independently of all other tasks that are currently running. Asynchronous tasks run in the background, so you can perform additional tasks while the asynchronous task is still running.

For example, to start a new shell and execute the UNIX **cp** command in that shell, you might use this statement:

```
systask command "cp /tmp/sas* ~/archive/" taskname="copyjob1"
                 status=copysts1 shell;
```

The return code from the **cp** command is saved in the macro variable COPYSTS1.

The output from the command is displayed in the SAS log.

Note: Program steps that follow the SYSTASK statements in SAS applications usually depend on the successful execution of the SYSTASK statements. Therefore, syntax errors in some SYSTASK statements will cause your SAS application to abort. △

There are two types of tasks that can be run with SYSTASK:

Task
All tasks started with SYSTASK COMMAND are of type Task. For these tasks, if you do not specify STATVAR or STATE, then SYSTASK LIST displays the task name, type, and state, and the name of the status macro variable. You can use SYSTASK KILL to kill only tasks of type Task.

SAS/Connect Process
: Tasks started from SAS/Connect with the RSUBMIT statement are of type SAS/Connect Process. For SAS/Connect processes, SYSTASK LIST displays the task name, type, and state. You cannot use SYSTASK KILL to kill a SAS/Connect process. For information on starting SAS/Connect processes with RSUBMIT, refer to *SAS/CONNECT User's Guide*.

The SYSRC macro variable contains the return code for the SYSTASK statement. The status variable that you specify with the STATUS option contains the return code of the process started with SYSTASK COMMAND. To ensure that a task executes successfully, you should monitor both the status of the SYSTASK statement and the status of the process that is started by the SYSTASK statement.

If a SYSTASK statement cannot execute successfully, the SYSRC macro variable will contain a non-zero value. For example, there may be insufficient resources to complete a task or the SYSTASK statement may contain syntax errors. With the SYSTASK KILL statement, if one or more of the processes cannot be killed, SYSRC is set to a non-zero value.

When a task is started, its status variable is set to NULL. You can use the status variables for each task to determine which tasks failed to complete. Any task whose status variable is NULL did not complete execution. If a task terminates abnormally, then its status variable will be set to `–1`. See "WAITFOR" on page 250 for more information about the status variables.

Unlike the X statement, you cannot use the SYSTASK statement to start a new interactive session.

See Also

- "WAITFOR" on page 250
- "X" on page 251
- "Executing Operating System Commands from Your SAS Session" on page 12

TITLE

Specifies title lines for SAS output

Valid: anywhere

UNIX specifics: maximum length of title

Syntax

TITLE <*n*> <'*text*' | "*text*">;

Details

In interactive modes, the maximum title length is 254 characters; otherwise, the maximum length is 200 characters. If the length of the specified title is greater than the value of the LINESIZE option, the title is truncated to the line size.

WAITFOR

Suspends execution of the current SAS session until the specified tasks finish executing

Valid: anywhere

UNIX specifics: all

Syntax

WAITFOR <_ANY | _ALL_> *taskname* <*taskname...*> <TIMEOUT=*seconds*>;

taskname
specifies the name of the task(s) that you want to wait for. See "SYSTASK" on page 246 for information about task names. The task name(s) that you specify must match exactly the task names assigned through the SYSTASK COMMAND statement. You cannot use wildcards to specify task names.

ANY | _ALL_
suspends execution of the current SAS session until either one or all of the specified tasks finishes executing. The default setting is _ANY_, which means that as soon as one of the specified task(s) completes executing, the WAITFOR statement will finish executing.

TIMEOUT=*seconds*
specifies the maximum number of seconds that WAITFOR should suspend the current SAS session. If you do not specify the TIMEOUT option, WAITFOR will suspend execution of the SAS session indefinitely.

Details

The WAITFOR statements suspends execution of the current SAS session until the specified task(s) finish executing or until the TIMEOUT interval (if specified) has elapsed. If the specified task was started with the XWAIT option, then the WAITFOR statement ignores that task. See "SYSTASK" on page 246 for a description of the XWAIT option.

For example, the following statements start three different SAS jobs and suspend the execution of the current SAS session until those three jobs have finished executing:

```
systask command "sas myprog1.sas" taskname=sas1;
systask command "sas myprog2.sas" taskname=sas2;
systask command "sas myprog3.sas" taskname=sas3;
waitfor _all_ sas1 sas2 sas3;
```

The SYSRC macro variable contains the return code for the WAITFOR statement. If a WAITFOR statement cannot execute successfully, the SYSRC macro variable will contain a non-zero value. For example, the WAITFOR statement may contain syntax errors. If the number of seconds specified with the TIMEOUT option elapses, then the WAITFOR statement finishes executing, and SYSRC is set to a non-zero value if

- you specify a single task that does not finish executing
- you specify more than one task and the _ANY_ option (which is the default setting), but none of the tasks finishes executing

- you specify more than one task and the _ALL_ option, and any one of the tasks does not finish executing.

Any task whose status variable is still NULL after the WAITFOR statement has executed did not complete execution. See "SYSTASK" on page 246 for a description of status variables for individual tasks.

See Also

- "SYSTASK" on page 246
- "X" on page 251
- "Executing Operating System Commands from Your SAS Session" on page 12

X

Issues an operating system command from within a SAS session

Valid: anywhere

UNIX specifics: valid operating system command

Syntax

X <'*host-command*'>;

host-command
specifies the UNIX command. If you specify only one UNIX command, you do not need to enclose it in quotes. Also, if you are running the SAS System from the Korn shell, you cannot use aliases.

Details

The X statement issues a UNIX command from within a SAS session. The SAS System executes the X statement immediately.

Neither the X statement nor the %SYSEXEC macro program statement is intended for use during the execution of a DATA step. The CALL SYSTEM routine, however, can be executed within a DATA step. See "CALL SYSTEM" on page 197 for an example.

Note: The X statement is not supported without arguments under the X Window System. △

See Also

- "Executing Operating System Commands from Your SAS Session" on page 12

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS® Companion for UNIX Environments, Version 8*, Cary, NC: SAS Institute Inc., 1999.

SAS® Companion for UNIX Environments, Version 8

ISBN 1–58025–502–7

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

1st printing, October 1999