CHAPTER

16

Macros

Macros in the CMS Environment

Most features of the SAS macro facility are portable. They are documented in *SAS Macro Language: Reference*. This section discusses only those components of the macro facility that have CMS-specific behavior.

Macro Variables

Portable Automatic Macro Variables That Have Host-Specific Values

The following automatic macro variables are portable, but their values are host-specific:

SYSDEVIC
contains the name of the current graphics device. The current graphics device is determined by the SAS system option DEVICE=. (See "DEVICE=" on page 266.) Ask your SAS Support Consultant which graphics devices are available at your site.

SYSENV
is provided for compatibility with the SAS System under other operating environments. Under CMS, its value is FORE when the TERMINAL system option is in effect; otherwise, its value is BACK. This is a read-only value.

SYSJOBID
gives the userid of the virtual machine that invoked the current SAS session.

SYSRC
contains the return code from the most recent operating environment command that was issued from within a SAS session.

SYSMAXLONG
returns the maximum long integer value allowed by CMS, which is 2,147,483,647.

SYSSCP
contains the operating environment abbreviation CMS.

SYSSCPL
contains the operating environment abbreviation VM/ESA. You cannot change the value of this variable.

Macro Statements

%KEYDEF
is analogous to the windowing environment KEYDEF command. It enables you to define function keys. The form of this statement is

%KEYDEF <'> *key-name*<'> <'*definition*'> ;

The number of keys available depends on your terminal. Most terminals that are used under CMS have either 12 or 24 program function (PF) keys. To define a key, specify the *key-name* (F1 through F24) of the key and the new *definition*.

If you omit the definition, SAS prints a message in the log that shows the current definition of the key; otherwise, the key's definition is changed to whatever you specified.

%CMS
executes CMS commands. It is similar to the CMS statement, which is described in "CMS" on page 221. The %CMS statement enables you to execute CMS commands immediately. It places the return code in the automatic variable &SYSRC. You can use the %CMS statement either inside or outside a macro. The form of the statement is

%CMS <*command*>;

You can use any CMS command or any sequence of macro operations that generate a CMS command. If you omit the *command*, your SAS session is suspended and your CMS session is placed in CMS subset mode. To return to the SAS session, type **return** and press Enter.

If a SAS program that contains a %CMS statement is transported to another operating environment, the %CMS statement is treated as a comment.

%SYSEXEC
executes CMS commands. The form of the statement is

%SYSEXEC <*command*>;

Under CMS, the %SYSEXEC statement works exactly like the %CMS statement. The two statements are different only if you transport your SAS program to a different operating environment. Because %SYSEXEC statements are recognized on multiple operating environments, each operating environment expects commands that are appropriate for that operating environment.

Macro Functions

%SCAN
under CMS and other environments that use the EBCDIC collating sequence, if you specify no delimiters, SAS treats all of the following characters as delimiters:

blank . < (+ | & ! $ *); ¬ – / , % ¦ ¢

%SYSGET
returns the value of operating environment variables and symbols from within a macro, in an interactive SAS session, or in open code. For information about using %SYSGET, see “Accessing System Variables” on page 10 .

Autocall Libraries

An autocall library contains files that define SAS macros. SAS Institute supplies some autocall macros in the system autocall library; you can also define autocall macros yourself in a user autocall library. In order to use an autocall library, the SAS system option MAUTOSOURCE must be in effect. (See *SAS Language Reference: Dictionary* for details about MAUTOSOURCE.)

Specifying the Autocall Library

The SASAUTOS= option specifies the autocall library. An autocall library contains files that define SAS macros. In order to use the autocall facility, the SAS system option MAUTOSOURCE must be in effect. The syntax of the SASAUTOS= option is

SASAUTOS=*file-specification* | (’*file-specification-1*’ . . . ’*file-specification-n*’)

You can use the following arguments with the SASAUTOS= option:

file-specification
specifies the name of an aggregate external file that contains the SAS autocall macros. Each member is used to hold the source statements for one macro. Member names must be the same as the name of the macro.

You can specify the aggregate in a number of ways; see “SASAUTOS=” on page 286 for details.

fileref
specifies a logical name that points to an aggregate external file.

Stored Compiled Macro Facility

The stored compiled macro facility gives you access to permanent SAS catalogs that contain compiled macros. The purpose of the stored compiled macro facility is to improve efficiency in the execution of production jobs. In order for SAS to use stored compiled macros, the SAS system option MSTORED must be in effect. In addition, you use the SAS system option SASMSTORE= to specify the libref of a SAS data library that contains a catalog of stored compiled SAS macros. For more information about these options, see "System Options in the CMS Environment" on page 252 and *SAS Language Reference: Dictionary*.

Using stored compiled macros offers the following advantages over other methods of making macros available to your session:

- SAS does not have to compile a macro definition when a macro call is made. Therefore, your program may be more efficient.
- Session-compiled macros and the autocall facility are also available in the same session.

Because you cannot re-create the source statements from a compiled macro, you must save the original macro source statements.

Using the stored compiled macro facility is the most efficient way to make macros accessible to your SAS session. However, you can use these approaches:

- place all macro definitions in the program before calling them
- use a %INCLUDE statement to bring macro definitions in the program from external files
- use the autocall facility to search predefined source libraries for macro definitions.

Accessing Stored Compiled Macros

The following example illustrates how to create a stored compiled macro in one session and then how to use the macro in a later session.

```
/* Create stored compiled macro */
libname mylib 'mylib a';
options mstored sasmstore=mylib;
%macro myfiles / store;
   filename file1 'first mylib a';
   filename file2 'second mylib a';
%mend;

/* Use stored compiled macro later */
libname mylib 'mylib a';
options mstored sasmstore=mylib;

%myfiles
data _null_;
   infile file1;
      ...statements reading input FILE1...
   file file2;
      ...statements writing output FILE2...
run;
```

Other Host-Specific Aspects of the Macro Facility

Collating Sequence for Evaluating Macro Characters

Under CMS, the macro facility uses the EBCDIC collating sequence for %EVAL and for implicit evaluation of macro characters.

SAS System Options Used by the Macro Facility

The following table lists the SAS system options that are used by the macro facility and that have host-specific characteristics. It also tells you where to look for more information about these system options.

Table 16.1 SAS System Options Used by the Macro Facility That Have Host-Specific Aspects

System Option	Description	See ...
MSYMTABMAX=	specifies the maximum amount of memory available to all symbol tables (global and local combined). The value of *n* can be expressed either as an integer or as MAX (the largest integer your operating environment can represent, typically 2,147,483,647). Under CMS, the default value for this option is 524,288 bytes.	*SAS Language Reference: Dictionary*
MVARSIZE=	specifies the maximum number of bytes for any macro variable stored in memory (0<=*n*<=32,768). The default setting for CMS is 8192.	See "MVARSIZE=" in the *System Options* section
SASAUTOS=	specifies the autocall library	See the information on SASAUTOS= in this section and in the *System Options* section

Customizing a Production Job

The SYSPARM= option supplies a value for the SYSPARM macro variable at SAS invocation. For example, to create a title based on a city as part of noninteractive execution the production program might contain the following statement:

```
TITLE "1996 Standard of Living Index for &sysparm";
```

The invocation command in CMS might be as follows:

```
sas program-name (sysparm=Boston)
```

Additional Sources of Information

- *SAS Macro Language: Reference*
- *SAS Macro Facility Tips and Techniques*
- *SAS Language Reference: Dictionary*

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

SAS® Companion for the CMS Environment, Version 8

ISBN 1–58025–481–0

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

1st printing, October 1999