CHAPTER 9

Storing and Reusing Macros

Introduction

When you submit a macro definition, by default, the macro processor compiles and stores the macro in a SAS catalog in the WORK library. These macros, referred to as *session compiled macros*, exist only during the current SAS session. To save frequently used macros between sessions, you can use either the autocall macro facility or the stored compiled macro facility.

The autocall macro facility stores the source for SAS macros in a collection of external files called an *autocall library*. The autocall facility is useful when you want to create a pool of easily maintained macros in a location that can be accessed by different applications and users. Autocall libraries can be concatenated together. The primary disadvantage of the autocall facility is that the first time that an autocall macro is called in a session, the macro processor compiles it. This compilation is overhead that you can avoid by using the stored compiled macro facility.

The stored compiled macro facility stores compiled macros in a SAS catalog in a SAS data library that you specify. By using stored compiled macros, you may save macro compilation time in your production-level jobs. However, because these stored macros are compiled, you must save and maintain the source for the macro definitions in a different location.

The autocall facility and the stored compiled macro facility each offer advantages. Some of the factors that determine how you choose to save a macro definition are how often you use a macro, how often you change it, how many users need to execute it, and how many compiled macro statements it has. If you are developing new programs, consider creating macros and compiling them during your current session. If you are running production-level jobs using name-style macros, consider using stored compiled macros. If you are allowing a group of users to share macros, consider using the autocall facility.

Note: For greater efficiency, store only name-style macros if you use the stored compiled macro facility. Storing statement-style and command-style macros is less efficient. △

It is good practice, when you are programming stored compiled macros or autocall macros, to use the %LOCAL statement to define macro variables that will be used only inside that macro. Otherwise, values of macro variables defined outside of the current macro might be altered. See the discussion of macro variable scope in Chapter 5, "Scope of Macro Variables."

Saving Macros in an Autocall Library

Generally, an autocall library is a directory containing individual files, each of which contains one macro definition. In Release 6.11 or later, an autocall library can also be a SAS catalog (see "Using SAS Catalogs as Autocall Libraries" below).

Operating Environment Information: Autocall Libraries on Different Hosts The term *directory* refers to an aggregate storage location that contains files (or members) managed by the host operating system. Different host operating systems identify an aggregate storage location with different names, such as a directory, a subdirectory, a maclib, a text library, or a partitioned data set. For more information, see the SAS Companion for your operating system. △

Using Directories as Autocall Libraries

To use a directory as a SAS autocall library, do the following:

1 To create library members, store the source code for each macro in a separate file in a directory. The name of the file must be the same as the macro name. For example, the statements defining a macro you would call by submitting %SPLIT must be in a file named SPLIT.

   Operating Environment Information: Autocall Library Member Names On operating systems that allow filenames with extensions, you must name autocall macro library members with a special extension, usually **`.SAS`**. Look at the autocall macros on your system provided by SAS Institute to determine whether names of files containing macros must have a special extension at your site.
   On MVS operating systems, you must assign the macro name as the name of the PDS member. △

2 Set the SASAUTOS system option to specify the directory as an autocall library. On most hosts, the reserved fileref SASAUTOS is assigned at invocation time to the autocall library supplied by SAS Institute or another one designated by your site. If you are specifying one or more autocall libraries, remember to concatenate the autocall library supplied by the Institute with your autocall libraries so that these macros will also be available. For details, refer to your host documentation and SASAUTOS in Chapter 13, "Macro Language Dictionary."

When storing files in an autocall library, remember the following:

□ Although the SAS System does not restrict the type of material you place in an autocall library, you should store only autocall library files in it to avoid confusion and for ease of maintenance.

□ Although the SAS System lets you include more than one macro definition, as well as open code, in an autocall library member, you should generally keep only one macro in any autocall library member. If you need to keep several macros in the same autocall library member, keep related macros together.

Using SAS Catalogs as Autocall Libraries

In Release 6.11 or later, you can use the CATALOG access method to store autocall macros as SOURCE entries in SAS catalogs. To create an autocall library using a SAS catalog, follow these steps:

1. Use a LIBNAME statement to assign a libref to the SAS library.
2. Use a FILENAME statement with the CATALOG argument to assign a fileref to the catalog that contains the autocall macros. For example, the following code creates a fileref, MYMACROS, that points to a catalog named MYMACS.MYAUTOS:

   ```
   libname mymacs 'SAS-data-library';
   filename mymacros catalog 'mymacs.myautos';
   ```

3. Store the source code for each macro in a SOURCE entry in a SAS catalog. (SOURCE is the entry type.) The name of the SOURCE entry must be the same as the macro name.
4. Set the SASAUTOS system option to specify the fileref as an autocall library. For more information, see SASAUTOS in Chapter 13.

Calling an Autocall Macro

To call an autocall macro, the system options MAUTOSOURCE must be set and SASAUTOS must be assigned. MAUTOSOURCE enables the autocall facility, and SASAUTOS specifies the autocall libraries. For more information on these options, see MAUTOSOURCE and SASAUTOS in Chapter 13.

Once you have set the required options, calling an autocall macro is like calling a macro that you have created in your current session. However, it is important that you understand how the macro processor locates the called macro. When you call a macro, the macro processor searches first for a session compiled macro definition. Next, the macro processor searches for a permanently stored compiled macro. If compiled stored macros are enabled with the MSTORED option, the macro processor opens the macro catalog in the library specified in the SASMSTORE option. If the macro processor does not find a compiled macro, and if MAUTOSOURCE is set, the macro processor opens libraries specified by the SASAUTOS option in the order in which they are specified in the option. It then searches each library for a member with the same name as the macro you invoked. When SAS finds a library member with that name, the macro processor does the following:

1. compiles all of the source statements in that member, including any and all macro definitions, and stores the result in the session catalog.
2. executes any open code (macro statements or SAS source statements not within any macro definition) in that member.
3. executes the macro within it with the name you invoked.

Note: If an autocall library member contains more than one macro, the macro processor compiles all of the macros but executes only the macro with the name you invoked. △

Any open code statements in the same autocall library member as a macro execute only the first time you invoke the macro. When you invoke the macro later in the same session, the compiled macro is executed, which contains only the compiled macro definition and not the other code the autocall macro source file may have contained.

It is not advisable to change SASAUTOS during a SAS session. If you change the SASAUTOS= specification in an ongoing SAS session, the SAS System will store the new specification only until you invoke an uncompiled autocall macro and then will close all opened libraries and open all the newly specified libraries that it can open.

For information about debugging autocall macros, see Chapter 10, "Macro Facility Error Messages and Debugging."

Saving Macros Using the Stored Compiled Macro Facility

The stored compiled macro facility compiles and saves compiled macros in a permanent catalog in a library that you specify. This compilation occurs only once. If the stored compiled macro is called in the current or later sessions, the macro processor executes the compiled code.

Compiling and Storing a Macro Definition

To compile a macro definition in a permanent catalog, create and save the source for each stored compiled macro. To store the compiled macro, follow these steps:

1 Use the STORE option in the %MACRO statement. Optionally, you can assign a descriptive title for the macro entry in the SAS catalog, by specifying the DES= option. For example, the %MACRO statement in the following definition shows the STORE and DES= options:

```
%macro myfiles / store
     des='Define filenames';
  filename file1 'external-file-1';
  filename file2 'external-file-2';
%mend;
```

CAUTION:

Save your macro source code. You cannot re-create the source statements from a compiled macro. Therefore, you must save the original macro source statements if you want to change the macro. For all stored compiled macros, you should document your macro source code well and save it. It is recommended that you save the source code in the same catalog as the compiled macro. In this example, save it to

```
mylib.sasmacro.myfiles.source
```

△

2 Set the MSTORED system option to enable the stored compiled macro facility. For more information, see MSTORED in Chapter 13.

3 Assign the SASMSTORE option to specify the SAS data library that contains or will contain the catalog of stored compiled SAS macros. For example, to store or call compiled macros in a SAS catalog named MYLIB.SASMACR, submit these statements.

```
libname mylib 'SAS-data-library';
options mstored sasmstore=mylib;
```

For more information, see SASMSTORE in Chapter 13.

4 Submit the source for each macro that you want to compile and permanently store.

You cannot move a stored compiled macro to another operating system. You can, however, move the macro source code to another operating system where you can then compile and store it. You may need to recompile these macros if you use them in a different release of the SAS System. For more information, see your host companion.

Storing Autocall Macros Supplied by SAS Institute

If you use the macros in the autocall library supplied by SAS Institute, you can save macro compile time by compiling and storing those macros in addition to ones you create yourself. Many of the macros related to base SAS software that are in the autocall library supplied by the Institute can be compiled and stored in a SAS catalog named SASMACR by using the autocall macro COMPSTOR that is supplied by SAS Institute. For more information, see COMPSTOR in Chapter 13.

Calling a Stored Compiled Macro

Once you have set the required system options, calling a stored compiled macro is just like calling session compiled macros. However, it is important that you understand how the macro processor locates a macro. When you call a macro, the macro processor searches for the macro name using this sequence:

1 the macros compiled during the current session
2 the stored compiled macros in the SASMACR catalog in the specified library (if options MSTORED and SASMSTORE= are in effect)
3 each autocall library specified in the SASAUTOS option (if options SASAUTOS= and MAUTOSOURCE are in effect).

You can display the entries in a catalog containing compiled macros. For more information, see Chapter 10.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS Macro Language: Reference, Version 8*, Cary, NC: SAS Institute Inc., 1999. 310 pages.

SAS Macro Language: Reference, Version 8

1-58025-522-1

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

1st printing, October 1999