STAT 541

Chapter 10: Storing Macro Programs and Advanced Macro Techniques

Reusing Macro Programs

- Macros in temporary SAS catalogs are only available for execution during the current SAS session. Such catalogs are deleted at the end of the session.
- Macros can be stored permanently for reuse later.
- Methods for storing macros permanently:
  - the % INCLUDE statement
  - the autocall macro facility
  - permanently stored compiled macros

Storing Macro Definitions in External Files

- Save macros in an external file.
- Use a %INCLUDE statement to insert the statements into a program.

%INCLUDE file-specification </SOURCE2>;

  - file-specification is the location of the file with the SAS code to be inserted
  - SOURCE2 directs SAS to display the inserted SAS code in the log

Advantage of Storing Macro Definitions in External Files and Using the %INCLUDE Statement

- Source code of the macro definition does not need to be in the program.
- A single copy of the macro definition is accessible to other programs.
- Macro definitions in external files are easily viewed and edited with any text editor.
- No special SAS system options are required to access macros this way.

Using the AUTOCALL Facility

- Permanently store macro definitions in source libraries called autocall libraries.
- An autocall library, whether it be the default one or a user-created one, is either a SAS catalog, an external directory, or a partitioned data set.
- SAS provides several macro definitions in a default autocall library.
- Multiple autocall libraries can be concatenated.
- Specify the SASAUTOS and MAUTOSOURCE system options.

Default Autocall Library & Autocall Macros Provided with SAS Software

- SAS provides several macros in a default autocall library.
- The libraries provided by SAS will depend on the SAS products licensed to your site.
- These autocall macros can be used without having to define or include them in your programs.
- Upon SAS installation, the autocall libraries are included in the value of the SASAUTOS system option in the configuration file.

Accessing Autocall Macros

- In order to access autocall macros, use two system options:
- MAUTOSOURCE system option must be specified
- SASAUTOS system options must identify location of autocall library or libraries

MAUTOSOURCE System Option Specifies Autocall Facility is Available

OPTIONS MAUTOSOURCE | NOMAUTOSOURCE;

- MAUTOSOURCE is the default and causes the macro processor to search the autocall libraries for a member with the requested name when a macro name is not found in the WORK library.
- NOMAUTOSOURCE prevents the macro processor from searching the autocall libraries when a macro name is not found in the WORK library.

SASAUTOS Controls Where Macro Facility Looks for Autocall Macros

Options SASAUTOS= *library-1;*

Options SASAUTOS= (*library-1,* . . . , *library-n*);

- *library-1* is a location that contains library members that contain a SAS macro definition. A location can be a SAS fileref or a host-specific location name enclosed in quotation marks. Each member contains a SAS macro definition.
- (*library-1,* . . . , *library-n*) identifies two or more locations that contain library members that contain a SAS macro definition. When you specify two or more autocall libraries, enclose the specifications in parentheses and separate them with either a comma or a blank space.

The DOSUBL Function

- The DOSUBL function takes as its input a text string, and enables immediate execution (in a "side session") of the SAS code within it.
- Macro variables that are created or updated during the code's execution are passed back to the calling environment.
- DOSUBL is used in a DATA step, or with %SYSFUNC outside a step.

More on DOSUBL

- DOSUBL can conveniently run a data-driven macro program that executes repeatedly for the various values of a variable in a data set (see example), without tediously calling the macro repeatedly.
- DOSUBL can be a convenient tool for simplifying a complicated program.

DOSUBL and CALL EXECUTE

- The DOSUBL function is similar to the CALL EXECUTE subroutine, but that has different timing: CALL EXECUTE waits until the DATA step is completed before executing the code.
- See examples to show the differences in how DOSUBL and CALL EXECUTE are processed by SAS.