# A Macro Library Facility For SAS[1] Software

Ross Z. Merlin, Pinkerton Computer Consultants Inc.

## ABSTRACT

To use a stored SAS macro requires two operations: the macro must be loaded (by %INCLUDE), then it must be invoked. Once a macro has been loaded, it remains available throughout the session and should not be reloaded; however, if many macros are being used, it is not reasonable to expect the user to keep track of which macros have been loaded and which have not. Other programming languages load macros only when needed -- why not the SAS system?

The primary purpose of this macro library facility is to let the software manage a collection of macros; i.e. to load macros only when necessary, to make macros readily available through a library, and to provide a framework for the standardization of parameter names.

One use of this facility is to maintain a library of "general-purpose" or "utility" macros -- a "programmer's toolbox"; another use is to build a simple command language or prompting system for non-programmer users so that they can take advantage of the power of the SAS system without having to learn the language.

## THE LIBRARY MANAGER

The heart of the system is a "loader" macro which serves as a pre-processor for the %INCLUDE statement. This macro checks to see if it has already loaded the requested macro. If the macro has already been loaded there is no need to execute the %INCLUDE statement again; the loader simply executes the requested macro. If the requested macro is not known to the loader, the %INCLUDE statement will be executed. The loader will then determine if the %INCLUDE was successful. A successful load will cause two actions: the macro's name will be added to the list of those known to the loader, and the requested macro will be executed. An unsuccessful load will cause an error message to be printed by a %PUT statement. Figure 1 exhibits the loader macro.

## DETAILS OF THE SYSTEM

The macros are stored in an OS PDS or a CMS MACLIB. There are two requirements for every member (except the loader): each member must end with the statement %LET LOADED=Y; (this must be outside the macro definition) and the member name must be identical to the macro name.

To assure that the loader is loaded will require two modifications to the SAS command (at installations where vendor software may not be modified, a copy of the SAS command can be modified and given a similar name such as SASX). The first modification is the addition of a statement to allocate the MACLIB (MACro LIBrary); the second modification is an INITSTMT specification to include the loader from the MACLIB. For example, if the loader's name is PLEASE:

```
ALLOC F(MACLIB) DA('XXX.MACLIB') SHR
SASCP TASKLIB(&TASKLIB &LOAD) -
  OPTIONS('&OPTIONS +
  INITSTMT=''%INC MACLIB(PLEASE);''')...
```

The second modification precludes the use of INITSTMT by the user, absent further modifications.

Positional and keyword parameters may be used, although parameters containing blanks will require special handling. The trick to parameter handling by the loader is that any parameter list passed to it is regarded as a single parameter (referred to by the loader as MACPARM). When the loader executes the requested macro (referred to by the loader as MACNAME), &MACPARM is resolved and the commas take their expected role as delimeters. This role change is possible because the loader is executed as a statement-style macro call, whereas the requested macro is executed as a name-style macro call. The macro exhibited in Figure 2 (a utility macro to find the number of observations in a SAS dataset) could be invoked by:

```
PLEASE OBS DATA=STATES,MACVAR=;
```

"PLEASE" is the name of the loader; "OBS" is the value of the loader's first parameter (MACNAME); "DATA=STATES,MACVAR=" is the value of the loader's second parameter (MACPARM). The string "DATA=STATES,MACVAR=" looks like two parameters, neither of which appears in the macro definition of the loader. Why does MACPARM take the entire string? For statement-style macros, "... in the macro call all values are separated by blanks instead of commas, and commas are treated as characters in the value instead of as delimeters."[2]

## IMPLEMENTATION

The name for the loader should be something that is easy to remember. Think of the loader's name as the "magic word". What's the magic word? It is PLEASE (Mother would be so proud!). Depending upon the environment in which this facility is implemented, the "magic word" could be the name or initials of the company (e.g. FORD or GM), an office or

function (e.g. SALES), or anything that seems relevant (e.g. POWER for an electric utility company).

For macros that will be shared by several programmers, parameter names should be standardized. For example, they can correspond to DATA step keywords and PROCedure options: FILE, INFILE, DATA, OUT, INDD, OUTDD, etc.

Each macro should include built-in HELP. For example, PLEASE OBS HELP=YES; prints the macro's purpose and lists the parameters. Any invocation of OBS that doesn't include HELP=YES will bypass the help display and perform the intended action.

There should be a HELP macro for the entire system, invoked by PLEASE HELP; . At the minimum, this should list the names of the macros in the MACLIB. Figure 3 exhibits the statements for a more extensive HELP macro; Figure 4 exhibits an execution of this macro.

Versions of this software have been used under MVS, SAS releases 79.5, 79.6, 82.3, and 82.4; and also under VM/CMS, SAS release 82.2. Several macro language bugs were encountered along the way, and ZAPs were required to correct the problems. An up-to-date version of the USAGE NOTES file[3] (including ZAPs) must be available, and the ZAPs recommended for the macro language should be applied.

## COSTS AND BENEFITS

When this facility is used option IMPLMAC is in effect, which means that "... the macro processor examines the first word of every statement to see if that word is a statement-style macro call."[4] Also, there is the cost of compiling the loader macro when the session begins. Finally, every macro invoked by this facility causes some execution of the loader macro.

A benefit of this system is that it avoids the unnecessary reloading of a macro; one such avoidance may more than offset the additional overhead, since "... the execution of a macro is less expensive than its definition."[5] Another benefit is the potential for increased programmer productivity. If programmers can be persuaded to pool their most useful macros (or to make selected programs available to others as generalized macros), programmers will not waste their time "reinventing the wheel".[6] Ideally, one person should be responsible for maintaining the integrity of the shared macro library: all macros in the library must be error-free, documented as to author and purpose, in conformance with local standards for parameter names, listed in the system HELP macro, and must have a built-in HELP feature.

For non-programmer users, there is the advantage of fewer syntax rules. The user needs neither the %INCLUDE statement nor the percent sign (and possibly the parentheses) in each macro invocation. If the HELP macro and the HELP=YES parameter are implemented, users can resolve many problems on their own.

## CONCLUSION

This software is used in an information retrieval system[7] that lets non-programmer users create their own reports from a database without coding a single SAS statement. A surprising variety of reports can be efficiently generated by the user. Properly designed macros with parameters can lead to efficient code generation, and this Macro Library Facility for SAS Software can lead to efficient operation of the application system by managing the macro library.

Please send questions or comments to:

Ross Z. Merlin
Pinkerton Computer Consultants, Inc.
5881 Leesburg Pike, Suite 400
Falls Church, VA 22041

or call (202) 632-0032

## NOTES

1) SAS is the registered trademark of SAS Institute Inc., Cary, NC 27511, USA.

2) Technical Report: P-127 Enhancements to the SAS Macro Facility, SAS Institute Inc., page 5.

3) SAS Communications, Volume VIII Number 4 (Spring 1983), SAS Institute Inc., page 21.

4) Technical Report: P-127 Enhancements to the SAS Macro Facility, SAS Institute Inc., page 13.

5) SAS User's Guide: Basics, 1982 edition, SAS Institute Inc., page 466.

6) Several utility macros are presented in "Statement Style Macros" by Donald J. Henderson and David L. Kuhn, SUGI Proceedings of the Ninth Annual Conference (SUGI '84), SAS Institute Inc., pages 979-988.

7) This paper is based on work performed under an Agency for International Development contract: AID/OTR-0000-C-00-4167-00.

```
OPTIONS IMPLMAC;
%GLOBAL MACLIST;
%MACRO PLEASE(MACNAME,MACPARM) / STMT;
    %LET LOADED=N;
    %LET MACNAME=%UPCASE(&MACNAME);
    %IF %LENGTH(&MACLIST)<=%LENGTH(&MACNAME) %THEN %LET WHERE=0;
    %ELSE %LET WHERE=%INDEX(&MACLIST,&MACNAME..);
    %IF &WHERE¬=0 %THEN %DO;
        %IF &MACPARM¬= %THEN %&MACNAME(&MACPARM);
        %ELSE %STR(%&MACNAME;);
    %END;
    %ELSE %DO;
       %INCLUDE MACLIB(&MACNAME);
       %IF &LOADED=Y %THEN %DO;
           %LET MACLIST=&MACLIST.&MACNAME..;
           %IF &MACPARM¬= %THEN %&MACNAME(&MACPARM);
           %ELSE %STR(%&MACNAME;);
       %END;
       %ELSE %PUT ERROR¬ PERHAPS THE MACRONAME WAS MISSPELLED;
   %END;
%MEND;
CLEAR;
%PUT AGENCY  FOR  INTERNATIONAL  DEVELOPMENT;
%PUT MACRO LIBRARY FACILITY FOR SAS SOFTWARE;
%PUT %STR(    FOR HELP, TYPE:   PLEASE HELP;);
%PUT %STR( );
```

Figure 1 -- the macro loader "PLEASE"

```
%MACRO OBS(DATA=_LAST_,FILE=LOG,MACVAR=NOBS,HELP=);
%IF %UPCASE(&HELP)=YES %THEN %DO;
%PUT "OBS" BY ROSS Z. MERLIN, 18DEC1984;
%PUT PURPOSE:  FIND # OF OBS IN A SAS DATASET.;
%PUT FEATURES: ANSWER MAY BE PRINTED TO A;
%PUT FILE AND/OR STORED IN A MACRO VARIABLE.;
%PUT PARAMETERS AND DEFAULTS --;
%PUT DATA=_LAST_    SASDATASET NAME;
%PUT FILE=LOG       WHERE TO PRINT ANSWER;
%PUT MACVAR=NOBS    NAME OF MACRO VARIABLE;
%PUT HELP=          HELP=YES FOR INFO;
%END;
%ELSE %DO;
DATA _NULL_;
    SET &DATA NOBS=NOBS POINT=POINT;
     _ERROR_=0;
    %IF &FILE¬= %THEN %DO;
        FILE &FILE;
        %IF %UPCASE(&DATA)=_LAST_ %THEN
         %LET DATA=&SYSDSN;
        PUT "&DATA HAS " NOBS " OBS.";
    %END;
    %IF &MACVAR¬= %THEN
     %STR(CALL SYMPUT("&MACVAR",PUT(NOBS,12.)););
    STOP;
   RUN;
%END;
%MEND OBS;
%LET LOADED=Y;
```

Figure 2 -- macro "OBS"

```
%MACRO HELP;
    %LOCAL MAC MACX;
    %PUT %STR( );
    %PUT THE FOLLOWING MACROS HAVE BEEN USED:;
    %LET MACX=1;
    %LET MAC=?;
    %DO%UNTIL(&MAC=);
        %LET MAC=%SCAN(&MACLIST,&MACX);
        %PUT %STR(     )&MAC;
        %LET MACX=%EVAL(&MACX+1);
    %END;
    %PUT %STR( );
    %PUT THE FOLLOWING UTILITY MACROS ARE AVAILABLE;
    %PUT COUNTRY  - FINDS COUNTRY NAME FOR A CODE AND VICE VERSA.;
    %PUT DECODE   - FINDS DATA ELEMENT CODE FOR ANY KEYWORD.;
    %PUT HELP     - SHOWS WHICH MACROS ARE AVAILABLE.;
    %PUT INDEXB   - BUILDS AN INDEX DATASET.;
    %PUT MAKEFMT  * MAKES A FORMAT FROM A SAS DATASET.;
    %PUT OBS      * SHOWS THE NUMBER OF OBSERVATIONS IN A DATASET.;
    %PUT PLEASE   - MACRO LOADER -- DO NOT TRY "%STR(PLEASE PLEASE;)";
    %PUT REPORTS  - ESDB REPORT GENERATOR.;
    %PUT SEARCH   - SEARCHES A SAS DATASET, GENERAL FORM OF "OECODE".;
    %PUT TYPE     * DETERMINE IF VARIABLE IS NUMERIC OR CHARACTER.;
    %PUT USAGE    - FSBROWSE SAS USAGE NOTES.;
    %PUT VARNAME  - SHOWS VARIABLES AND LABELS IN A SAS DATASET.;
    %PUT WP       * WORD PROCESSOR.;
    %PUT * INDICATES MORE HELP AVAILABLE BY EXECUTING DESIGNATED;
    %PUT %STR(  )MACRO FOLLOWED BY HELP=YES;
    %PUT %STR( );
    %PUT FOR ADDITIONAL INFO, CONTACT ROSS Z. MERLIN, (202)632-0032;
    %PUT %STR( );
%MEND HELP;
%LET LOADED=Y;
```

Figure 3 -- macro "HELP"

```
 THE FOLLOWING MACROS HAVE BEEN USED:
      OBS
      VARNAMES
      COUNTRY
      SEARCH
      HELP

 THE FOLLOWING UTILITY MACROS ARE AVAILABLE
 COUNTRY  - FINDS COUNTRY NAME FOR A CODE AND VICE VERSA.
 DECODE   - FINDS DATA ELEMENT CODE FOR ANY KEYWORD.
 HELP     - SHOWS WHICH MACROS ARE AVAILABLE.
 INDEXB   - BUILDS AN INDEX DATASET.
 MAKEFMT  * MAKES A FORMAT FROM A SAS DATASET.
 OBS      * SHOWS THE NUMBER OF OBSERVATIONS IN A DATASET.
 PLEASE   - MACRO LOADER -- DO NOT TRY "PLEASE PLEASE;"
 REPORTS  - ESDB REPORT GENERATOR.;
 SEARCH   - SEARCHES A SAS DATASET, GENERAL FORM OF "DECODE".
 TYPE     * DETERMINE IF VARIABLE IS NUMERIC OR CHARACTER.
 USAGE    - FSBROWSE SAS USAGE NOTES.
 VARNAME  - SHOWS VARIABLES AND LABELS IN A SAS DATASET.
 WP       * WORD PROCESSOR.
 * INDICATES MORE HELP AVAILABLE BY EXECUTING DESIGNATED
   MACRO FOLLOWED BY HELP=YES

 FOR ADDITIONAL INFO, CONTACT ROSS Z. MERLIN, (202)632-0032
```

Figure 4 -- execution of the "HELP" macro