Paper 126-29

Using Macros to Automate SAS® Processing

Kari Richardson, SAS Institute, Cary, NC
Eric Rossland, SAS Institute, Dallas, TX

ABSTRACT

This hands-on workshop shows how to use the SAS Macro Facility to automate a SAS job. In this hands-on workshop you build a basic macro to process a data set. You then expand the macro using SAS Input/Output functions to allocate a SAS library and read various items from the descriptor portions of the data sets in that library. You further enhance the macro to conditionally process the desired data sets in the selected library and send a summary report via e-mail.

INTRODUCTION

The SAS Macro Facility is a tool for extending and customizing the SAS System and for reducing the amount of text you must enter to do common tasks. This paper and the corresponding hands-on workshop introduce the basics of the SAS Macro Facility and show how it can be used to automate and extend a SAS program. This hands-on demo shows how macros can be written to automate the processing of data sets that contain employee expenses.

SETUP

The setup process for this workshop includes an autoexec program, which allocates a SAS library reference named SUGI. Two programs are then run to create the data for the workshop.

The CREATE_DATA.SAS program is run to generate the course data. The output of the program shows that six data sets were created:

Output - (Untitled)

The SAS System

The CONTENTS Procedure

-----Directory-----

Libref: SUGI
Engine: V8
Physical Name: C:\workshop\ws126
File Name: C:\workshop\ws126

#	Name	Memtype	File Size	Last Modified
1	EXPENSES_BANGOR	DATA	5120	23JAN2004:10:05:07
2	EXPENSES_CHICAGO	DATA	5120	23JAN2004:10:05:07
3	EXPENSES_DALLAS	DATA	5120	23JAN2004:10:05:07
4	EXPENSES_MIAMI	DATA	5120	23JAN2004:10:05:07
5	EXPENSES_NEWYORK	DATA	5120	23JAN2004:10:05:07
6	EXPENSES_PHOENIX	DATA	5120	23JAN2004:10:05:07

The CREATE_FORMAT.SAS program is run to create a permanent format that will be used when processing the newly created data sets.

Output - (Untitled)

The SAS System

FORMAT NAME: $CATMAX LENGTH: 3 NUMBER OF VALUES: 4
MIN LENGTH: 1 MAX LENGTH: 40 DEFAULT LENGTH 3 FUZZ: 0

START	END	LABEL (VER. 8.2 23JAN2004:10:06:53)
Entertainment	Entertainment	50
Lodging	Lodging	200
Meal	Meal	50
Transportation	Transportation	45

EXPENSE1 MACRO

Each of the expense report data sets is currently processed by running an existing program, PROCESSDATA.SAS. The first macro created, EXPENSE1, will be used to automate the processing of these data sets. The macro contains two required parameters: the physical path of the SAS data library and the name of a data set to be processed. The macro then

- allocates a libref named SUGI
- verifies that the libref was allocated successfully
- verifies the existence of the specified data set
- conditionally reads and validates the data set if it exists or ends execution if the data set does not exist.

PROCESSDATA.SAS

```
options fmtsearch=(sugi) missing=' ';
data valid invalid;
  set sugi.&ds;
  if amount le input(put(category,$catmax.),3.) then output valid;
  else output invalid;
run;
ods listing close;
ods html file='reports.htm';
proc tabulate data=valid format=dollar10.2;
  class empid category;
  var amount / style=[background=gray foreground=white font_style=italic];
  keylabel sum=' ' all='Total';
  keyword all / style=[background=gray foreground=white font_style=italic];
  title 'Employee Expense Summary by Category';
  table empid all*[style=[background=gray foreground=white
                          font_weight=bold font_style=italic]],
        category*amount={label=''} all*amount*[style=[background=gray
                                                      foreground=white
                                                      font_weight=bold
                                                      font_style=italic]];
run;
proc print data=invalid noobs n='Number of Invalid Expense Items= ' label;
  sum amount;
  format amount dollar10.2;
  title '***Invalid Expenses***';
run;
ods html close;
ods listing;
```

BEGIN MACRO DEFINITION AND SPECIFY POSITIONAL PARAMETERS

The %MACRO statement begins the macro definition and assigns a name to the macro. In the macro below, the statement also specifies two positional parameters. The %MEND statement is used to end the macro definition.

```
%macro expense1(lib,ds);
%mend expense1;
```

EXECUTE THE LIBNAME STATEMENT AND VERIFY THE LIBREF WAS ASSIGNED

The LIBNAME statement is used to assign the libref SUGI to the directory location provided by the first parameter, LIB.

```
libname sugi "&lib";
```

The SAS LIBREF function is used to verify if a libref has been allocated. The LIBREF function returns a zero if the specified libref has been assigned and a non-zero value if the libref has not been assigned. The %SYSFUNC macro function executes SAS functions. The %GOTO statement branches macro processing to the specified label. **Caution:** When using the %GOTO statement, do not use a % before the label name. Using a % before the label name tells the macro processor to call a macro with that name to generate the label text.

```
%if %sysfunc(libref(sugi)) ne 0 %then %do;
   %put ERROR:  The SAS Data Library could;
   %put ERROR-  not be established;
   %goto quit;
%end;
```

VERIFY THE EXISTENCE OF THE SPECIFIED DATA SET

The SAS EXIST function verifies that a data library member exists. By default the EXIST function checks for a member type of DATA. The EXIST function returns a 1 if the library member exists and a zero if the member name does not exist or if the member type is invalid.

```
%if %sysfunc(exist(sugi.&ds))=0 %then %do;
  %if %sysfunc(exist(sugi.&ds,view))=0 %then
   %do;
     %put ERROR:  The data set &ds does not;
     %put ERROR-  exist in the specified;
     %put ERROR-  location (&lib).;
     %goto quit;
   %end;
%end;
```

CONDITIONALLY PROCESS THE DATA SET OR END EXECUTION OF THE MACRO

The %INCLUDE statement retrieves SAS source code from an external file. The SOURCE2 option causes the inserted SAS statements to appear in the SAS log. The PROCESSDATA.SAS program contains a DATA step and two PROC steps to read and process the specified data set and create two reports. %QUIT is a macro label; in this macro, the %QUIT is right before the %MEND. When the %GOTO statement branches macro processing to the %QUIT label, the macro execution stops.

```
   %include "c:\workshop\ws126\processdata.sas" / source2;
   %quit:
%mend expense1;
```

TEST THE MACRO

Test the macro using these three scenarios:

- provide a valid data set name (expenses_dallas)
  `%expense1(.,expenses_dallas)`
- provide an invalid data set name (expenses_montreal)
  `%expense1(.,expenses_montreal)`
- provide a non-existent directory location (c:\kari) and an invalid data set name (testit).
  `%expense1(c:\kari,testit)`

The following is the SAS output expected from the test:

Employee Expense Summary by Category

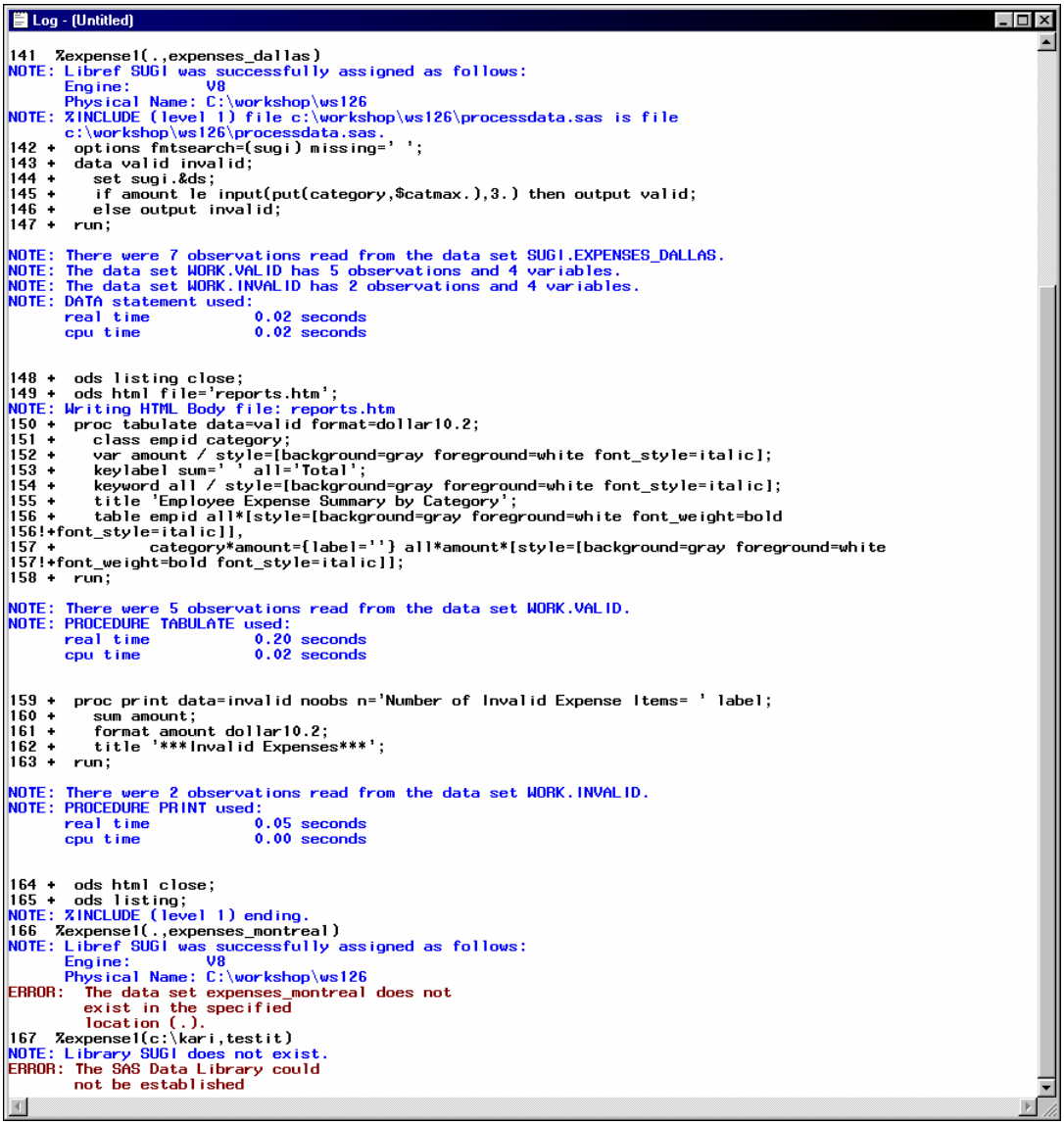
	Expense Category			Total
	Lodging	Meal	Transportation	Expense Amount
Employee ID				
E1524	$60.32	$37.85	$34.50	$132.67
E1970	$111.66	$45.06		$156.72
Total	$171.98	$82.91	$34.50	$289.39

****Invalid Expenses****

Employee ID	Date of Expense	Expense Category	Expense Amount
E1524	10MAR2004	Meal	$52.64
E1970	07APR2004	Transportation	$46.08
			$98.72

Number of Invalid Expense Items= 2

The following is the SAS log expected from the test:



```
Log - (Untitled)

141  %expense1(.,expenses_dallas)
NOTE: Libref SUGI was successfully assigned as follows:
      Engine:        V8
      Physical Name: C:\workshop\ws126
NOTE: %INCLUDE (level 1) file c:\workshop\ws126\processdata.sas is file
      c:\workshop\ws126\processdata.sas.
142 +  options fmtsearch=(sugi) missing=' ';
143 +  data valid invalid;
144 +    set sugi.&ds;
145 +    if amount le input(put(category,$catmax.),3.) then output valid;
146 +    else output invalid;
147 +  run;

NOTE: There were 7 observations read from the data set SUGI.EXPENSES_DALLAS.
NOTE: The data set WORK.VALID has 5 observations and 4 variables.
NOTE: The data set WORK.INVALID has 2 observations and 4 variables.
NOTE: DATA statement used:
      real time           0.02 seconds
      cpu time            0.02 seconds


148 +  ods listing close;
149 +  ods html file='reports.htm';
NOTE: Writing HTML Body file: reports.htm
150 +  proc tabulate data=valid format=dollar10.2;
151 +    class empid category;
152 +    var amount / style=[background=gray foreground=white font_style=italic];
153 +    keylabel sum=' ' all='Total';
154 +    keyword all / style=[background=gray foreground=white font_style=italic];
155 +    title 'Employee Expense Summary by Category';
156 +    table empid all*[style=[background=gray foreground=white font_weight=bold
156!+font_style=italic]],
157 +          category*amount={label=''} all*amount*[style=[background=gray foreground=white
157!+font_weight=bold font_style=italic]];
158 +  run;

NOTE: There were 5 observations read from the data set WORK.VALID.
NOTE: PROCEDURE TABULATE used:
      real time           0.20 seconds
      cpu time            0.02 seconds


159 +  proc print data=invalid noobs n='Number of Invalid Expense Items= ' label;
160 +    sum amount;
161 +    format amount dollar10.2;
162 +    title '***Invalid Expenses***';
163 +  run;

NOTE: There were 2 observations read from the data set WORK.INVALID.
NOTE: PROCEDURE PRINT used:
      real time           0.05 seconds
      cpu time            0.00 seconds


164 +  ods html close;
165 +  ods listing;
NOTE: %INCLUDE (level 1) ending.
166  %expense1(.,expenses_montreal)
NOTE: Libref SUGI was successfully assigned as follows:
      Engine:        V8
      Physical Name: C:\workshop\ws126
ERROR:  The data set expenses_montreal does not
        exist in the specified
        location (.).
167  %expense1(c:\kari,testit)
NOTE: Library SUGI does not exist.
ERROR: The SAS Data Library could
       not be established
```

EXPENSE1 MACRO DEFINITION

The complete EXPENSE1 macro definition is listed below:

```
%macro expense1(lib,ds);
  libname sugi "&lib";
  %if %sysfunc(libref(sugi)) ne 0 %then %do;
    %put ERROR: The SAS Data Library could;
    %put ERROR- not be established;
    %goto quit;
  %end;
  %if %sysfunc(exist(sugi.&ds)) = 0 %then %do;
    %if %sysfunc(exist(sugi.&ds,view)) = 0 %then %do;
     %put ERROR:  The data set &ds does not;
     %put ERROR-  exist in the specified;
     %put ERROR-  location (&lib).;
      %goto quit;
    %end;
  %end;
  %include "c:\workshop\ws126\processdata.sas" / source2;
  %quit:
%mend expense1;
```

EXPENSE2 MACRO

The EXPENSE1 macro automates processing of the data sets but must be invoked for each one. The second macro created, EXPENSE2, will be used to process all the data sets in a specified library. The macro contains one required parameter: the physical path of the SAS data library. The macro then

- allocates a libref named SUGI
- verifies that the libref was allocated successfully
- creates a macro variable containing the list of all data sets in the specified library
- processes all data sets and separates valid and invalid expenses
- creates an HTML report for both valid and invalid expenses by including the REPORTS.SAS program.

REPORTS.SAS

```
options missing=' ';
proc tabulate data=valid format=dollar10.2;
  class empid category;
  var amount / style=[background=gray foreground=white font_style=italic];
  keylabel sum=' ' all='Total';
  keyword all / style=[background=gray foreground=white font_style=italic];
  title 'Employee Expense Summary by Category';
  table empid all*[style=[background=gray foreground=white
                          font_weight=bold font_style=italic]],
        category*amount={label=''} all*amount*[style=[background=gray
                                                      foreground=white
                                                      font_weight=bold
                                                      font_style=italic]];
run;
proc print data=invalid noobs n='Number of Invalid Expense Items= ' label;
  sum amount;
  format amount dollar10.2;
  title '***Invalid Expenses***';
run;
```

BEGIN MACRO DEFINITION AND SPECIFY A POSITIONAL PARAMETER

```
%macro expense2(lib);
%mend expense2;
```

EXECUTE THE LIBNAME STATEMENT AND VERIFY THAT THE LIBREF WAS ASSIGNED

This process is identical to the EXPENSE1 macro.

```
libname sugi "&lib";
%if %sysfunc(libref(sugi)) ne 0 %then %do;
   %put ERROR:  The SAS Data Library could;
   %put ERROR-  not be established;
   %goto quit;
%end;
```

CREATE A LIST OF ALL DATA SETS IN THE SPECIFIED LIBRARY

A DATA _NULL_ step is used to create a macro variable containing a list of all the data sets in the SUGI library. The SASHELP.VSTABLE dictionary table is used to determine the names of each data set in the SUGI library. These names are concatenated together in a DATA step variable. The CALL SYMPUT routine is then used to create a new macro variable based on the DATA step variable.

```
options fmtsearch=(sugi) missing=' ';
data _null_;
   length datasets $200; retain datasets;
   set sashelp.vstable end=eof;
   where libname='SUGI';
   datasets=trim(datasets)||
            ' '||trim(libname)||
            '.'||trim(memname);
   if eof then call symput('datasets',datasets);
run;
```

PROCESS ALL DATA SETS AND SEPARATE VALID AND INVALID EXPENSES

The next step is to read all the data sets in the SUGI library and determine if each observation from each data set is a valid expense. This will be accomplished using the $CATMAX. format that was created during the setup step and stored in the SUGI library. Because this is a permanent format, the FMTSEARCH option is used to add the SUGI library to the format search path.

```
options fmtsearch=(sugi);
```

The $CATMAX. format contains the maximum amount allowed for each expense category. If the expense item is more than the maximum allowed, then that observation is considered invalid. The DATA step below creates two temporary data sets that contain the valid and invalid expense items.

```
data valid invalid;
  set &datasets;
  if amount<input(put(category,$catmax.),3.)
    then output valid;
  else output invalid;
run;
```

CREATE AN HTML REPORT

After the valid and invalid data sets have been created, they can be processed to create an HTML report. The SAS Output Delivery System (ODS) is used to create an HTML report from the PROC TABULATE and PROC PRINT steps in the REPORT.SAS program which is included with the %INCLUDE statement.

```
    ods listing close;
    ods html file='report.htm';
    %include "c:\workshop\ws126\reports.sas" / source2;
    ods html close;
    ods listing;
    %quit:
%mend expense2;
```

TEST THE MACRO

Test the macro using these two scenarios:

- provide a valid SAS data library location (.)

```
%expense2(.)
```

- provide an invalid SAS data library location (c:\kari)

```
%expense2(c:\kari)
```

The following is the SAS output expected from the test:

Employee Expense Summary by Category

	Expense Category				Total
	Entertainment	Lodging	Meal	Transportation	Expense Amount
Employee ID					
E1524		$60.32	$37.85	$34.50	$132.67
E1970		$111.66	$45.06		$156.72
E2048		$136.27	$31.59	$18.82	$186.68
E2181		$164.24		$16.81	$181.05
E2333		$182.55			$182.55
E2940		$177.07	$38.62		$215.69
E3216			$38.82		$38.82
E3407		$152.15	$20.55	$26.42	$199.12
E3500		$31.21			$31.21
E3572	$10.47	$384.29	$47.91		$442.67
E3771	$25.97		$37.91		$63.88

E3811	$20.80				*$20.80*
E3910		$124.97	$25.19		*$150.16*
E4096			$49.97	$68.30	*$118.27*
E4362		$483.47	$31.03		*$514.50*
E4434			$16.54		*$16.54*
E4542	$44.90	$252.55			*$297.45*
E4666			$40.62	$9.91	*$50.53*
E4748		$76.14			*$76.14*
E4880			$44.72	$13.86	*$58.58*
E4953			$51.76		*$51.76*
E5619			$33.76	$61.74	*$95.50*
E5678			$82.68		*$82.68*
E5685		$118.54	$19.73		*$138.27*
E5692			$64.70		*$64.70*
E5792				$31.30	*$31.30*
E5832	$37.16			$33.66	*$70.82*
E5846		$300.18	$53.44	$12.27	*$365.89*
E5850		$138.28		$35.23	*$173.51*
E5951		$135.58		$42.02	*$177.60*
E6144				$37.11	*$37.11*
E6820		$111.22		$9.51	*$120.73*
Total	*$139.30*	*$3,140.69*	*$812.45*	*$451.46*	*$4,543.90*

******Invalid Expenses******

Employee ID	Date of Expense	Expense Category	Expense Amount
E6820	13APR2004	Transportation	$45.80
E2048	19MAR2004	Meal	$50.27
E2181	21APR2004	Entertainment	$68.51
E2181	30APR2004	Entertainment	$128.74
E1524	10MAR2004	Meal	$52.64
E1970	07APR2004	Transportation	$46.08
E4096	07MAR2004	Entertainment	$112.94
E4277	24APR2004	Entertainment	$93.13
E4277	20APR2004	Entertainment	$100.46
E4434	11MAR2004	Entertainment	$88.78
E4748	14MAR2004	Entertainment	$127.74
E3500	25MAR2004	Entertainment	$120.59
E3771	30MAR2004	Entertainment	$118.10
E3910	19APR2004	Entertainment	$114.72

E5619	27APR2004	Entertainment	$65.42
E5678	14APR2004	Entertainment	$131.13
E5685	31MAR2004	Entertainment	$51.68
E5792	28MAR2004	Entertainment	$110.97
E5846	09MAR2004	Entertainment	$98.49
E5846	17MAR2004	Entertainment	$107.55
E5846	10APR2004	Lodging	$206.90
E5951	07MAR2004	Entertainment	$96.27
E5951	05MAR2004	Transportation	$47.76
			$2,184.67
Number of Invalid Expense Items= 23			

The following is the SAS log expected from the test:

```
Log - (Untitled)
34   %expense2(.)
NOTE: Libref SUGI was successfully assigned as follows:
      Engine:        V8
      Physical Name: C:\workshop\ws126

NOTE: There were 6 observations read from the data set SASHELP.VSTABLE.
      WHERE libname='SUGI';
NOTE: DATA statement used:
      real time           1.04 seconds
      cpu time            0.19 seconds



NOTE: There were 4 observations read from the data set SUGI.EXPENSES_BANGOR.
NOTE: There were 13 observations read from the data set SUGI.EXPENSES_CHICAGO.
NOTE: There were 7 observations read from the data set SUGI.EXPENSES_DALLAS.
NOTE: There were 26 observations read from the data set SUGI.EXPENSES_MIAMI.
NOTE: There were 20 observations read from the data set SUGI.EXPENSES_NEWYORK.
NOTE: There were 33 observations read from the data set SUGI.EXPENSES_PHOENIX.
NOTE: The data set WORK.VALID has 80 observations and 4 variables.
NOTE: The data set WORK.INVALID has 23 observations and 4 variables.
NOTE: DATA statement used:
      real time           0.04 seconds
      cpu time            0.04 seconds


NOTE: Writing HTML Body file: reports.htm
NOTE: %INCLUDE (level 1) file reports.sas is file C:\workshop\ws126\reports.sas.
35   +
36   +  options missing=' ';
37   +
38   +  proc tabulate data=valid format=dollar10.2;
39   +     class empid category;
40   +     var amount / style=[background=gray foreground=white font_style=italic];
41   +     keylabel sum=' ' all='Total';
42   +     keyword all / style=[background=gray foreground=white font_style=italic];
43   +     title 'Employee Expense Summary by Category';
44   +     table empid all*[style=[background=gray foreground=white font_weight=bold
44 !+font_style=italic]],
45   +           category*amount={label=''} all*amount*[style=[background=gray foreground=white
45 !+font_weight=bold font_style=italic]];
46   +  run;

NOTE: There were 80 observations read from the data set WORK.VALID.
NOTE: PROCEDURE TABULATE used:
      real time           0.21 seconds
      cpu time            0.04 seconds


47   +  proc print data=invalid noobs n='Number of Invalid Expense Items= ' label;
48   +     sum amount;
49   +     format amount dollar10.2;
50   +     title '***Invalid Expenses***';
51   +  run;

NOTE: There were 23 observations read from the data set WORK.INVALID.
NOTE: PROCEDURE PRINT used:
      real time           0.08 seconds
      cpu time            0.02 seconds


NOTE: %INCLUDE (level 1) ending.
52   %expense2(c:\kari)
NOTE: Library SUGI does not exist.
ERROR: The SAS Data Library could
       not be established
```

EXPENSE2 MACRO DEFINITION

The complete EXPENSE2 macro definition is listed below:

```
%macro expense2(lib);
  libname sugi "&lib";
  %if %sysfunc(libref(sugi)) ne 0 %then %do;
    %put ERROR: The SAS Data Library could;
    %put ERROR- not be established;
    %goto quit;
  %end;
  options fmtsearch=(sugi) missing=' ';
  data _null_;
    length datasets $200; retain datasets;
    set sashelp.vstable end=eof;
    where libname='SUGI';
    datasets=trim(datasets)||
             ' '||trim(libname)||
             '.'||trim(memname);
    if eof then call symput('datasets',datasets);
  run;
  data valid invalid;
    set &datasets;
    if amount le input(put(category,$catmax.),3.) then output valid;
    else output invalid;
  run;
  ods listing close;
  ods html file='reports.htm';
        %include "reports.sas" / source2;
  ods html close;
  ods listing;
  %quit:
%mend expense2;
```

EXPENSE3 MACRO

The third macro created, EXPENSE3, will also process all data sets in a specified library but will additionally create a summary report of the processing completed, which is sent via email. The macro contains three keyword parameters: the physical path of the SAS data library, the email address to send a summary report to, and a physical location where the HTML report should be saved. The macro then

- allocates a libref named SUGI
- verifies that the libref was allocated successfully
- creates a macro variable containing the list of all data sets in the specified library
- processes all data sets and separates valid and invalid expenses
- creates a macro variable that has the total number of expense reports processed
- creates an HTML report for both valid and invalid expenses and stores it in the specified location
- creates a summary report and emails it to the specified e-mail address.

BEGIN MACRO DEFINITION AND SPECIFY KEYWORD PARAMETERS

This macro has keyword parameters. A keyword parameter is a parameter that can be assigned a default value. When the macro is called, a value can be specified for the keyword parameter or left off, in which case the default value is used.

```
%macro expense3(lib=c:\workshop\ws126,
                emailID=joe.smith@sugi.com,
                rptloc=c:\workshop\ws126);
%mend expense3;
```

EXECUTE THE LIBNAME STATEMENT AND VERIFY THAT THE LIBREF WAS ASSIGNED

This process is identical to the EXPENSE1 and EXPENSE2 macros.

```
libname sugi "&lib";
%if %sysfunc(libref(sugi)) ne 0 %then %do;
   %put ERROR:  The SAS Data Library could;
   %put ERROR-  not be established;
   %goto quit;
%end;
```

CREATE A LIST OF ALL DATA SETS IN THE SPECIFIED LIBRARY

This process is identical to the EXPENSE2 macro.

```
data _null_;
   length datasets $200; retain datasets;
   set sashelp.vstable end=eof;
   where libname='SUGI';
   datasets=trim(datasets)||' '||trim(libname)||'.'||trim(memname);
   if eof then call symput('datasets',datasets);
run;
```

PROCESS ALL DATA SETS, SEPARATE VALID AND INVALID EXPENSES, COUNT THE NUMBER OF EXPENSE REPORTS

This process is similar to the EXPENSE2 macro except for the additional step of accumulating the total number of expense reports processed across all data sets (NUMEXPRPTS) and storing it in a new macro variable (which is also called NUMEXPRPTS).

```
options fmtsearch=(sugi) missing=' ';
data valid invalid;
  drop numexprpts;
  set &datasets end=eof;
  if amount<input(put(category,$catmax.),3.) then output valid;
  else output invalid;
  numexprpts+1;
  if eof then call symput('numexprpts',trim(left(numexprpts)));
run;
```

CREATE AN HTML REPORT AND STORE IT IN THE SPECIFIED LOCATION

This process is similar to the EXPENSE2 macro except for where the HTML report will be stored, which is now provided by the keyword parameter RPTLOC.

```
ods listing close;
ods html file="&rptloc\report.htm";
%include "c:\workshop\ws126\reports.sas" / source2;
ods html close;
ods listing;
```

CREATE A SUMMARY REPORT AND E-MAIL IT TO THE SPECIFIED EMAIL ADDRESS

Using a FILENAME statement with the EMAIL device type creates a fileref that can be used to send an e-mail message. Depending on the type of e-mail system and the options specified, an attachment can also be added as show below. A DATA _NULL_ step is used here to create the text of the e-mail and send it to the MAILTO fileref.

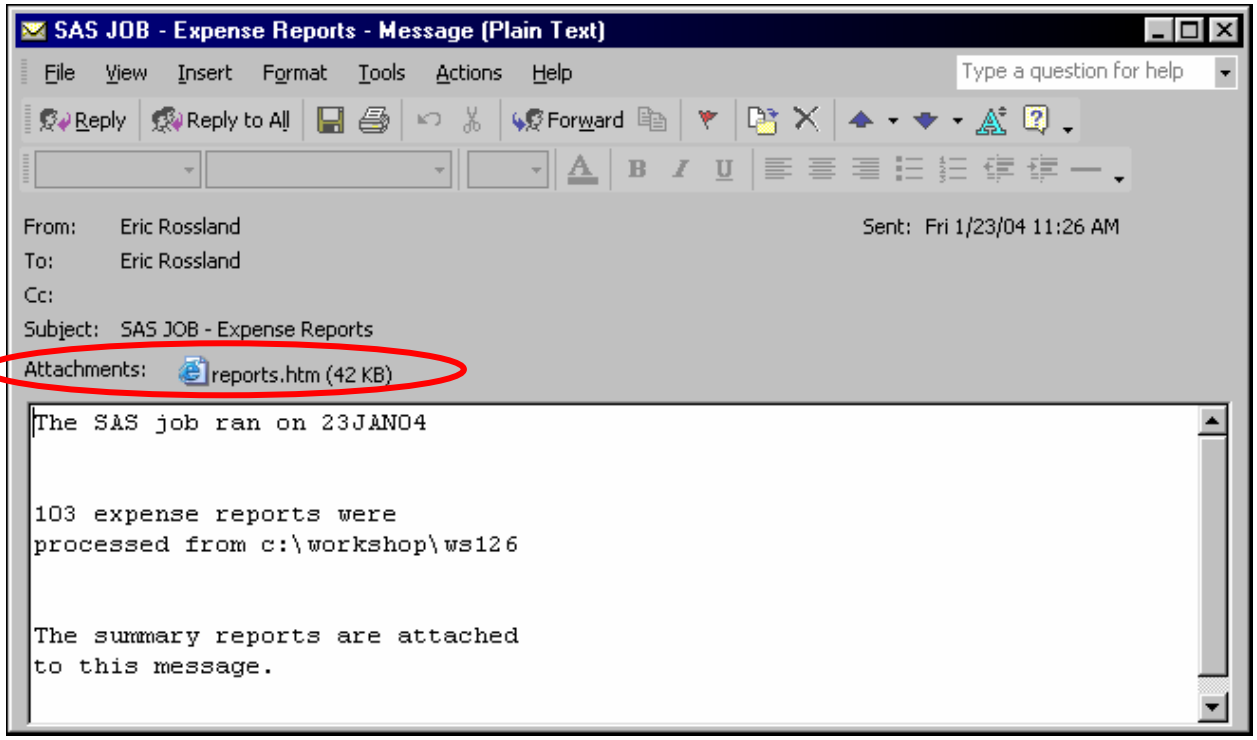
```
  filename mailto email "&emailID"
                  subject="SAS JOB - Expense Reports"
                  attach="&rptloc\reports.htm";
  data _null_;
     file mailto;
     put "The SAS job ran on &sysdate" //;
     put "&numexprpts expense reports were ";
     put "processed from &lib" //;
     put "The summary reports are attached ";
     put "to this message.";
  run;
  %quit:
%mend expense3;
```

TEST THE MACRO

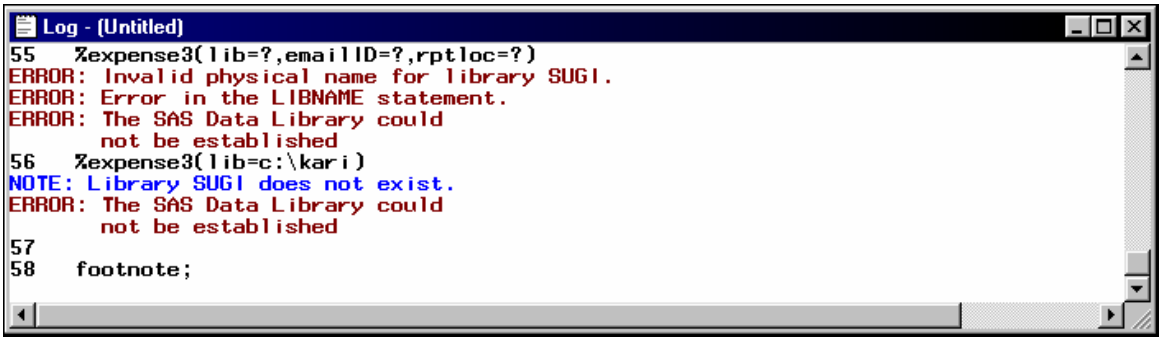
Test the macro using these three scenarios:

- provide no parameters (use the default values in the macro)
  `%expense3()`
- provide three invalid parameters
  `%expense3(lib=?,emailID=?,rptloc=?)`
- provide only one parameter and make it invalid
  `%expense3(lib=c:\kari)`

The following is the e-mail expected from the first test. Notice the attachment:



The following is the SAS log expected from the 2nd and 3rd tests (no e-mail was sent):



EXPENSE3 MACRO DEFINITION

The complete EXPENSE3 macro definition is listed below:

```
%macro expense3(lib=c:\workshop\ws126,
                emailID=joe.smith@sugi.com,
                rptloc=c:\workshop\ws126);
  libname sugi "&lib";
  %if %sysfunc(libref(sugi)) ne 0 %then %do;
    %put ERROR: The SAS Data Library could;
    %put ERROR- not be established;
    %goto quit;
  %end;
  data _null_;
     length datasets $200; retain datasets;
     set sashelp.vstable end=eof;
     where libname='SUGI';
     datasets=trim(datasets)||
              ' '||trim(libname)||
              '.'||trim(memname);
     if eof then
        call symput('datasets',datasets);
  run;
  options fmtsearch=(sugi) missing=' ';
  data valid invalid;
     drop numexprpts;
     set &datasets end=eof;
     if amount<input(put(category,$catmax.),3.) then output valid;
     else output invalid;
     numexprpts+1;
     if eof then call symput('numexprpts',trim(left(numexprpts)));
  run;
  ods listing close;
  ods html file="&rptloc\reports.htm";
    %include "c:\workshop\ws126\reports.sas" / source2;
  ods html close;
  ods listing;
  filename mailto email "&emailID" subject='SAS JOB - Expense Reports'
                                   attach="&rptloc\reports.htm";
  data _null_;
    file mailto;
    put "The SAS job ran on &sysdate" //;
    put "&numexprpts expense reports were ";
    put "processed from &lib" //;
    put "The summary reports are attached";
    put "to this message.";
  run;
  %quit:
%mend expense3;
```

FINAL ENHANCEMENT – DOCUMENT VALID EXPENSES

The PROC PRINT report lists all observations with invalid expense items. Adding a footnote to the report is one way to list which expenses are valid. A macro can be created to read the $CATMAX. format and create the footnote. This macro can then be added anywhere the footnote is desired, including the EXPENSE3 macro.

BEGIN MACRO DEFINITION AND SPECIFY KEYWORD PARAMETERS

```
%macro footnotes(libref=sugi,format=$catmax);
%mend footnotes;
```

DETERMINE VALID EXPENSES USING THE $CATMAX. FORMAT

The CNTLOUT= option of PROC FORMAT is used to create a data set that contains the values in the format.

```
proc format lib=&libref cntlout=work.temp;
   select &format;
run;
```

GENERATE MACRO VARIABLES FOR EACH FORMAT VALUE

The DATA _NULL_ step is used to create two series of macro variable: the unformatted information (the START variable) and the corresponding formatted value (the LABEL variable).

```
data _null_;
   set temp nobs=numobs;
   if _n_=1 then call symput('n',numobs);
   call symput('exptype'||trim(left(put(_n_,2.))),trim(start));
   call symput('value'||trim(left(put(_n_,2.))),trim(label));
run;
```

GENERATE THE FOOTNOTE STATEMENTS

Because the macro will be used in HTML reports, HTML tabs can be used to specify formatting attributes. The value for FOOTNOTE1 does not change.

```
footnote1 "<font face='Arial, Helvetica, Helv' size='4' color=blue>
           <b>The maximum values for expenses are as follows:</font>";
```

The other FOOTNOTE statements need to reflect the values stored in the format and are built dynamically using the macro variables created in the previous step. The macro variable N contains the number of formatted values and is used to create as many footnotes as there are formatted values. When &i=1, HTML tags are created for an HTML table. Each time through the loop, the unformatted information and corresponding formatted value is added to the HTML table. When &i=&n, the HTML tags to close the table are created.

```
%do i=1 %to &n;
   %let text=;
   %if &i=1 %then %do;
       %let text=<center><table>;
       %end;
   %let text=&text. <center><tr><td><font color=blue><b>&&exptype&i
                    </td><td align=right><font color=blue><b>&&value&i</td></tr>;
   %if &i=&n %then %do;
       %let text=&text. </table></font>;
       %end;
       footnote%eval(&i+1) font=Arial j=left "&text";
%end;
```

CLEAN UP

As a final step, after the %DO block, run a PROC DATASETS step to delete the data set WORK.TEMP that was created during this macro by PROC FORMAT.

```
proc datasets lib=work nolist;
   delete temp;
   run;
quit;
```

TEST THE MACRO

The FOOTNOTES macro can be called any time valid expense values should appear as a footnote. The following macro invocation calls the FOOTNOTES macro and uses the default values for the keyword parameters:

```
%footnotes();
```

The following is the expected footnote created from the test:

FOOTNOTES MACRO DEFINITION

The complete FOOTNOTES macro definition is listed below:

```
options mprint;
%macro footnotes(libref=sugi,format=$catmax);
   %local i text;
   proc format lib=&libref cntlout=work.temp;
     select &format;
   run;
   data _null_;
      set temp nobs=numobs;
      if _n_=1 then call symput('n',numobs);
      call symput('exptype'||trim(left(put(_n_,2.))),trim(start));
      call symput('value'||trim(left(put(_n_,2.))),trim(label));
   run;
   footnote1 "<font face='Arial, Helvetica, Helv' size='4' color=blue><b>
              The maximum values for expenses are as follows:</font>";
   %do i=1 %to &n;
      %let text=;
      %if &i=1 %then %do;
          %let text=<center><table>;
      %end;
      %let text=&text. <center><tr><td><font color=blue><b>&&exptype&i
                </td><td align=right><font color=blue><b>&&value&i</td></tr>;
      %if &i=&n %then %do;
          %let text=&text. </table></font>;
      %end;
      footnote%eval(&i+1) font=Arial j=left "&text";
   %end;
   proc datasets lib=work nolist;
      delete temp;
      run;
   quit;
%mend;
```

CONCLUSION

This hands-on workshop provided an overview of the SAS Macro Facility and a chance to work with some of the macro statements, routines, and functionality. Although this workshop used a specific data scenario, the information presented here and the techniques learned can be applied to many types of program automation.

REFERENCES

Additional information about the SAS Macro Facility can be found in the *SAS Macro Language: Reference, Version 8* or by attending the *SAS Macro Language* or *SAS Macro Programming: Advanced Topics* training courses.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Kari Richardson
SAS Institute Inc.
SAS Circle
Cary, NC 27513
E-mail: kari.richardson@sas.com

Eric Rossland
SAS Institute Inc.
15455 N. Dallas Parkway
Suite 1300
Addison, TX 75001
E-mail: eric.rossland@sas.com