MACROs - A FEW USES TO GET YOU STARTED

MICHAEL M. BROWN, SYNTEX RESEARCH

ABSTRACT

Due to the fact that MACROs are a large, complex, self-contained language, within an ever growing software system, figuring out where and how to use them can be difficult. Also, since there are always novice programmers appearing each year, reiterating MACRO uses becomes necessary. Therefore, this paper will present four ways to put the MACRO language to work. The first example shows how to create a block of statements for inclusion in other areas of the job. The second example, demonstrates a repetitive variable replacement within a block of statements. The third example shows how to use the %LET statement in the PROC step. Last, a use for the SYMGET statement will be demonstrated. The examples presented attempt to do so in a simple straightforward fashion; and when put to use they can become powerful.

INTRODUCTION

MACROs are very powerful and can be used in a very complicated manner, however, there are a few ways to use MACROs that are not so complicated. For instance, the same block of statements can be used over and over again. MACROS are good for simplifying this kind of problem. Instead of writing statements over and over, a programmer can write the statements once and embed them within a MACRO.

EXAMPLE 1, REPEATING BLOCKS OF CODE

This MACRO contains a DATA statement and a PROC PRINT statement which is executed before the data is statistically analyzed. The MACRO is named EX1, and will be stored temporarily in the program, by using the %MACRO EX1 and %MEND statements before and after the statements to be included in this particular MACRO. After creating this MACRO, it can be used later in the program by placing the statement %EX1 where this MACRO is to appear and be executed.

Figure 1 shows the DATA and MACRO EX1. Figure 2 shows program execution of this MACRO, and Figure 3 shows some of the output that was subsequently created.

EXAMPLE 2 USING MACROs WITH VARIABLE REPLACEMENT

MACRO usage is also very helpful when you have to run a PROC on one or two variables at a time, but you have quite a few variables to process. Instead of writing the same block of statements and changing the variable being processed, one block of MACRO statements is used to identify all my variables for replacement.

My second MACRO example does this. This MACRO contains a PROC TTEST and two MACRO variables in the VAR statement which will be used for replacement. The MACRO is named EX2 and the variables VAR1 and VAR2 are identified in parenthesis. After the MEND statement, the variables that are to be included in the PROC TTEST are identified with the MACRO replacement statement. This statement can be written as many times as one needs and will be executed that many times, so be careful.

Figure 4 shows the MACRO EX2 program statements, Figure 5 shows the program execution of this MACRO, and Figure 6 shows some of the output that was subsequently created.

EXAMPLE 3 USING THE %LET STATEMENT

Occasionally, a programmer may come upon a situation where it is necessary to pass a variable value to a group of MACRO statements. The %LET statement is good for this function.

In my third example of MACRO uses, the %LET statement is used to pass a value of a MACRO variable called CODE to the inside of a MACRO group. This MACRO group, CODCHART, uses the %IF, %THEN, and %DO statements to conditionally execute a PROC CHART, based on the value of the variable CODE. This example also shows how to use MACROs in a fashion that emulates DATA step programming.

Figure 7 shows the program statements using the %LET statement, Figure 8 shows the conditionally executed MACRO group, Figure 9 shows the output of the conditionally executed MACRO PROC group.

EXAMPLE 4 USING THE SYMGET STATEMENT

My final example shows one more way in which MACROs can be used without too much trouble. An occasion may arise where you want to assign a value of one MACRO variable to another variable. This can be done using the SYMGET statement.

In the final example, the SYMGET statement is used to retrieve the value of the variable CODE, which was used in the last DATA step. The SYMGET statement then passes the value of the MACRO variable CODE to the variable REPORT. This is used to execute a small report writer based on the value of the variable REPORT.

Figure 10 shows the statements used to pass the value of the MACRO variable CODE to the variable REPORT. Figure 11 shows the output produced by this sample used of SYMGET.

This MACRO program took some sales data and produced statistics, graphs and reports using some different MACRO statements. The way the MACROs were used was not too difficult to understand. Perhaps, you can use some of these examples or your own in a setting suitable for you.

I would like to acknowledge Marie Whalen and Kathy Jones for their valuable help in putting this poster/paper together.

wp/3443h

```
OPTIONS MACROGEN MPRINT SYMBOLGEN DQUOTE NONOTES;          *OPTIONS THAT HELP DEBUG MACRO CODE

DATA EXAMPS;
  INPUT SUBJECT GROUP $ DM01 DM02 DM03 DM04 DM05
  DM06 DM07 DM08;
CARDS;
 1  A  76  87  90  32  45  54  75  27
 2  B  65  76  82  45  65  87  91  91
 3  B  32  45  72  73  64  62  65  87
 4  B  27  44  52  71  63  74  91  87
 5  A  43  26  82  35  72  48  26  48
 6  A  25  36  47  58  40  58  46  95
 7  B  17  31  37  48  95  60  74  60
 8  B  52  47  26  69  85  85  73  59
 9  A  52  54  45  38  39  37  91  23
10  B  51  23  61  72  26  45  91  10
11  A  68  49  48  39  20  43  91  81
12  A  25  36  47  58  69  60  71  82

**EXAMPLE 1 BUILDING BLOCKS OF MACRO CODE;

%MACRO EX1;                                                *CREATION OF MACRO EX1
DATA EXAMPS;                                                 (PROC PRINT TO BE USED AGAIN)
 SET EXAMPS;

 PROC PRINT;
  VAR DM01-DM08;
  ID SUBJECT;
  %MEND;

  **THE MACRO GROUP WILL BE INCLUDED HERE;

%EX1;                                                      *INCLUDE THE DATASET
PROC FREQ;                                                   PRINT THE DATA BEFORE I DO A PROC FREQ
 TABLES DM01-DM08;

  **THE MACRO GROUP WILL BE INCLUDED HERE;

%EX1;                                                      *INCLUDE THE DATASET
PROC MEANS;                                                  PRINT THE DATA BEFORE I DO A PROC MEANS
 VAR DM01-DM08;

  **THE MACRO GROUP WILL BE INCLUDED HERE;

%EX1;                                                      *INCLUDE THE DATASET
PROC UNIVARIATE;                                             PRINT THE DATA BEFORE I DO A PROC UNIVARIATE
 VAR DM01-DM08;

**EXAMPLE 2 MACROS USING VARIABLE REPLACEMENT;
```

FIGURE 1 (MACRO EX1 PROGRAM STATEMENTS)

```
OPTIONS MACROGEN MPRINT SYMBOLGEN DQUOTE NONOTES;

DATA EXAMPS;
  INPUT SUBJECT GROUP $ DM01 DM02 DM03 DM04 DM05
  DM06 DM07 DM08;
CARDS;

**EXAMPLE 1 BUILDING BLOCKS OF MACRO CODE;

%MACRO EX1;
DATA EXAMPS;
 SET EXAMPS;

PROC PRINT;
 VAR DM01-DM08;
 ID SUBJECT;
 %MEND;

   **THE MACRO GROUP WILL BE INCLUDED HERE;

   %EX1
 + DATA EXAMPS;
 + SET EXAMPS;
 + PROC PRINT;
 + VAR DM01-DM08;
 + ID SUBJECT;

   PROC FREQ;
    TABLES DM01-DM08;

   **THE MACRO GROUP WILL BE INCLUDED HERE;

   %EX1
 + DATA EXAMPS;
PAGE 2 FOR DM01
PAGE 2 FOR DM02
PAGE 2 FOR DM03
PAGE 3 FOR DM04
PAGE 3 FOR DM05
PAGE 3 FOR DM06
PAGE 4 FOR DM07
PAGE 4 FOR DM08
 + SET EXAMPS;
 + PROC PRINT;
 + VAR DM01-DM08;
 + ID SUBJECT;

   PROC MEANS;
    VAR DM01-DM08;
 + ID SUBJECT;

   PROC UNIVARIATE;
    VAR DM01-DM08;
   **THE MACRO GROUP WILL BE INCLUDED HERE;

  %EX1
 + DATA EXAMPS;
 + SET EXAMPS;
 + PROC PRINT;
 + VAR DM01-DM08;
 + ID SUBJECT;

   PROC UNIVARIATE;
    VAR DM01-DM08
```

←INPUT THE DATA

←CREATED MACRO EX1

←INCLUDED MACRO EX1→
BEFORE PROC FREQ

```
1-%EX1
1-%EX1
1-%EX1
1-%EX1
1-%EX1
```

←PROC FREQ PAGES

←INCLUDED MACRO EX1→
BEFORE PROC FREQ

```
1-%EX1
1-%EX1
1-%EX1
1-%EX1
```

←INCLUDED MACRO EX1→
BEFORE PROC UNIVARIATE

```
1-%EX1
1-%EX1
1-%EX1
1-%EX1
1-%EX1
```

FIGURE 2 (MACRO EX1 EXECUTED PROGRAM STATEMENTS)

FIRST PROC PRINT→

PROC FREQ OUTPUT→

SAS

SUBJECT	DM01	DM02	DM03	DM04	DM05	DM06	DM07	DM08
1	76	87	90	12	45	54	75	27
2	65	76	82	45	65	87	91	91
3	32	45	12	73	64	67	65	87
4	21	44	52	71	61	14	91	87
5	43	26	82	35	72	48	26	48
6	25	36	47	58	40	58	48	95
7	17	71	37	48	95	60	74	95
8	52	47	26	69	85	85	73	59
9	52	54	45	38	39	37	91	23
10	51	23	61	72	26	45	91	10
11	68	49	48	39	20	43	91	81
12	25	36	47	58	69	60	71	82

SAS

DM01	FREQUENCY	PERCENT	CUMULATIVE FREQUENCY	CUMULATIVE PERCENT
17	1	8.3	1	8.3
21	1	8.3	2	16.7
25	2	16.7	4	33.3
32	1	8.3	5	41.7
43	1	8.3	6	50.0
51	1	8.3	7	58.3
52	2	16.7	9	75.0
65	1	8.3	10	83.3
68	1	8.3	11	91.7
76	1	8.3	12	100.0

DM02	FREQUENCY	PERCENT	CUMULATIVE FREQUENCY	CUMULATIVE PERCENT
23	1	8.3	1	8.3
26	1	8.3	2	16.7
36	2	16.7	4	33.3
44	1	8.3	5	41.7
45	1	8.3	6	50.0
47	1	8.3	7	58.3
49	1	8.3	8	66.7
54	1	8.3	9	75.0
73	1	8.3	10	83.3
76	1	8.3	11	91.7
87	1	8.3	12	100.0

DM03	FREQUENCY	PERCENT	CUMULATIVE FREQUENCY	CUMULATIVE PERCENT
12	1	8.3	1	8.3
26	1	8.3	2	16.7
37	1	8.3	3	25.0
45	1	8.3	4	33.3
47	2	16.7	6	50.0
48	1	8.3	7	58.3
52	1	8.3	8	66.7
61	1	8.3	9	75.0
82	2	16.7	11	91.7
90	1	8.3	12	100.0

FIGURE 3 (OUTPUT FROM MACRO EX1)

SAS

SECOND PROC PRINT→

SUBJECT	DMO1	DMO2	DMO3	DMO4	DMO5	DMO6	DMO7	DMO8
1	76	87	90	12	45	54	75	27
2	65	76	82	45	65	87	91	91
3	32	45	12	73	64	62	65	87
4	21	44	52	71	61	14	91	87
5	43	26	82	35	72	48	26	48
6	25	36	47	58	40	58	48	95
7	17	71	37	48	95	60	74	95
8	52	47	26	69	85	85	73	59
9	52	54	45	38	39	37	91	23
10	51	23	61	72	26	45	91	10
11	68	49	48	39	20	43	91	81
12	25	36	47	58	69	60	71	82

SAS

PROC MEANS OUTPUT____→

VARIABLE	N	MEAN	STANDARD DEVIATION	MINIMUM VALUE	MAXIMUM VALUE	STD ERROR OF MEAN	SUM	VARIANCE	CV
DMO1	12	43.91666667	19.86984162	17.00000000	76.00000000	5.73592920	527.00000000	394.81060606	45.244
DMO2	12	49.50000000	19.69540786	23.00000000	87.00000000	5.68557452	594.00000000	387.90909091	39.789
DMO3	12	52.41666667	23.23578092	12.00000000	90.00000000	6.70759219	629.00000000	539.90151515	44.329
DMO4	12	51.50000000	18.77861453	12.00000000	73.00000000	5.42091908	618.00000000	352.63636364	36.463
DMO5	12	56.75000000	22.98665621	20.00000000	95.00000000	6.63567607	681.00000000	528.38636364	40.505
DMO6	12	54.41666667	19.83320601	14.00000000	87.00000000	5.72535341	653.00000000	393.35606061	36.447
DMO7	12	73.91666667	20.26397759	26.00000000	91.00000000	5.84970646	887.00000000	410.62878788	27.415
DMO8	12	62.50000000	29.53734154	10.00000000	95.00000000	8.52669605	750.00000000	872.45454545	47.260

FIGURE 3 (OUTPUT FROM MACRO EX1)

SAS

THIRD PROC PRINT→

SUBJECT	DMO1	DMO2	DMO3	DMO4	DMO5	DMO6	DMO7	DMO8
1	76	87	90	12	45	54	75	27
2	65	76	82	45	65	87	91	91
3	32	45	12	73	64	62	65	87
4	21	44	52	71	61	14	91	87
5	43	26	82	35	72	48	26	48
6	25	36	47	58	40	58	48	95
7	17	71	37	48	95	60	74	95
8	52	47	26	69	85	85	73	59
9	52	54	45	38	39	37	91	23
10	51	23	61	72	26	45	91	10
11	68	49	48	39	20	43	91	81
12	25	36	47	58	69	60	71	82

SAS
UNIVARIATE

VARIABLE=DMO1

MOMENTS

N	12	SUM WGTS	12
MEAN	43.9167	SUM	527
STD DEV	19.8698	VARIANCE	394.811
SKEWNESS	0.153089	KURTOSIS	-1.32251
USS	27487	CSS	4342.92
CV	45.2444	STD MEAN	5.73593
T:MEAN=0	7.65642	PROB>|T|	0.0001
SGN RANK	39	PROB>|S|	0.00250684
NUM ^= 0	12		

QUANTILES(DEF=4)

100% MAX	76	99%	76
75% Q3	61.75	95%	76
50% MED	47	90%	73.6
25% Q1	25	10%	18.2
0% MIN	17	5%	17
		1%	17
RANGE	59		
Q3-Q1	36.75		
MODE	25		

EXTREMES

LOWEST	HIGHEST
17	52
21	52
25	65
25	68
32	76

PROC UNIVARIATE OUTPUT_____→

FIGURE 3 (OUTPUT FROM MACRO EX1)

```
%MACRO EX2 (VAR1, VAR2);

 PROC TTEST;
  CLASS GROUP;
  VAR &VAR1 &VAR2;

  %MEND;

%EX2 (DMO1, DMO2);
%EX2 (DMO3, DMO4);
%EX2 (DMO5, DMO6);
%EX2 (DMO7, DMO8);
```

←STATEMENTS CREATING MACRO EX2
(VAR1 & VAR2 ARE REPLACEMENT VARIABLES)

←STATEMENT IDENTIFYING VARIABLES THAT WILL BE USED FOR VAR1 AND VAR2. (THE BLOCK WILL BE EXECUTED 4 TIMES ONCE FOR EACH STATEMENT

FIGURE 4 (MACRO EX2 PROGRAM STATEMENTS)

```
**EXAMPLE 3 USING THE MACRO TO COMBINE DATA AND PROC STEPS;

 DATA EXAMPS3;
  SET EXAMPS;

%LET CODE=C;                                    <-THIS STATEMENT GIVES A VALUE TO THE VARIABLE CODE

%MACRO CODCHART;
%IF &CODE=A %THEN %DO;                          <-A GROUP OF %IF STATEMENTS THAT WILL BE EXCUTED
  PROC CHART;                                      CONDITIONALLY
  VBAR DM01;
  %END;
%ELSE %IF &CODE=B %THEN %DO;
  PROC CHART;
  VBAR DM02;
  %END;
%ELSE %IF &CODE=C %THEN %DO;
  PROC CHART;
  VBAR DM03;
  %END;
%ELSE %IF &CODE=D %THEN %DO;
  PROC CHART;
  VBAR DM04;
  %END;
%ELSE %IF &CODE=E %THEN %DO;
  PROC CHART;
  VBAR DM05;
  %END;
%ELSE %IF &CODE=F %THEN %DO;
  PROC CHART;
  VBAR DM06;
  %END;
%ELSE %IF &CODE=G %THEN %DO;
  PROC CHART;
  VBAR DM07;
  %END;
%ELSE %IF &CODE=H %THEN %DO;
  PROC CHART;
  VBAR DM08;
  %END;
%ELSE %IF &CODE=I %THEN %DO;
  PROC CHART;
  VBAR DM09;
  %END;
%ELSE %IF &CODE=J %THEN %DO;
  PROC CHART;
  VBAR DM010;
  %END;
%ELSE %IF &CODE=K %THEN %DO;
  PROC CHART;
  VBAR DM011;
  %END;
%ELSE %IF &CODE=L %THEN %DO;
  PROC CHART;
  VBAR DM012;
  %END;
%MEND CODCHART;

%CODCHART;
```

FIGURE 7 (MACRO CODCHART PROGRAM STATEMENTS)

```
      %MACRO EX2 (VAR1,VAR2);                   <-MACRO BLOCK NAMED EX2

       PROC TTEST;
        CLASS &GROUP;
        VAR &VAR1 &VAR2;

      %MEND;

       %EX2 (DM01,DM02)                         <-STATEMENTS CREATED FROM          1-%EX1
     + PROC TTEST;                                 THE REPLACEMENT VARIABLES      1-%EX1
     + CLASS &GROUP;                                                              1-%EX1
     + VAR DM01 DM02;

       %EX2 (DM03,DM04)                                                           1-%EX1
     + PROC TTEST;                                                                1-%EX1
     + CLASS &GROUP;                                                              1-%EX1
     + VAR DM03 DM04

       %EX2 (DM05,DM06)                                                           1-%EX1
     + PROC TTEST;                                                                1-%EX1
     + CLASS &GROUP;                                                              1-%EX1
     + VAR DM05 DM06;

       %EX2 (DM07,DM08)                                                           1-%EX1
     + PROC TTEST;                                                                1-%EX1
     + CLASS &GROUP;                                                              1-%EX1
     + VAR DM07 DM08;
```

FIGURE 5 (MACRO EX2 EXECUTED PROGRAM STATEMENTS)

```
SAS (R) LOG  OS SAS 5.16              MVS/XA JOB

 DATA EXAMPS3;
  SET EXAMPS;

%LET CODE=C;

%MACRO CODCHART;
%IF &CODE=A %THEN %DO;
  PROC CHART;
  VBAR DM01;
  %END;
%ELSE %IF &CODE=B %THEN %DO;
  PROC CHART;
  VBAR DM02;
  %END;
%ELSE %IF &CODE=C %THEN %DO;
  PROC CHART;
  VBAR DM03;
  %END;
%ELSE %IF &CODE=D %THEN %DO;
  PROC CHART;
  VBAR DM04;
  %END;
%ELSE %IF &CODE=E %THEN %DO;
  PROC CHART;
  VBAR DM05;
  %END;
%ELSE %IF &CODE=F %THEN %DO;
  PROC CHART;
  VBAR DM06;
  %END;
%ELSE %IF &CODE=G %THEN %DO;
  PROC CHART;
  VBAR DM07;
  %END;
%ELSE %IF &CODE=H %THEN %DO;
  PROC CHART;
  VBAR DM08;
  %END;
%ELSE %IF &CODE=I %THEN %DO;
  PROC CHART;
  VBAR DM09;
  %END;
%ELSE %IF &CODE=J %THEN %DO;
  PROC CHART;
  VBAR DM010;
  %END;
%ELSE %IF &CODE=K %THEN %DO;
  PROC CHART;
  VBAR DM011;
  %END;
%ELSE %IF &CODE=L %THEN %DO;
  PROC CHART;
  VBAR DM012;
  %END;
%MEND CODCHART;

%CODCHART;
+ PROC CHART;
+ VBAR DM03;
```

←THIS IS THE CONDITION THAT GETS EXECUTED BASED ON THE VALUE OF THE VARIABLE CODE

FIGURE 8 (MACRO CODCHART EXECUTED PROGRAM STATEMENTS)

SAS

TTEST PROCEDURE

VARIABLE: DM01

GROUP	N	MEAN	STD DEV	STD ERROR	MINIMUM	MAXIMUM	VARIANCES	T	DF	PROB > |T|
A	6	48.16666667	21.36742068	8.72321297	25.00000000	76.00000000	UNEQUAL	0.7248	9.9	0.4854
B	6	39.66666667	19.20069443	7.83865068	17.00000000	65.00000000	EQUAL	0.7248	10.0	0.4852

FOR HO: VARIANCES ARE EQUAL, F'= 1.24 WITH 5 AND 5 DF PROB > F'= 0.8202

VARIABLE: DM02

GROUP	N	MEAN	STD DEV	STD ERROR	MINIMUM	MAXIMUM	VARIANCES	T	DF	PROB > |T|
T										
A	6	48.00000000	21.58703314	8.81206938	26.00000000	87.00000000	UNEQUAL	-0.2523	9.9	0.8059
B	6	51.00000000	19.54482029	7.97913947	23.00000000	76.00000000	EQUAL	-0.2523	10.0	0.8059

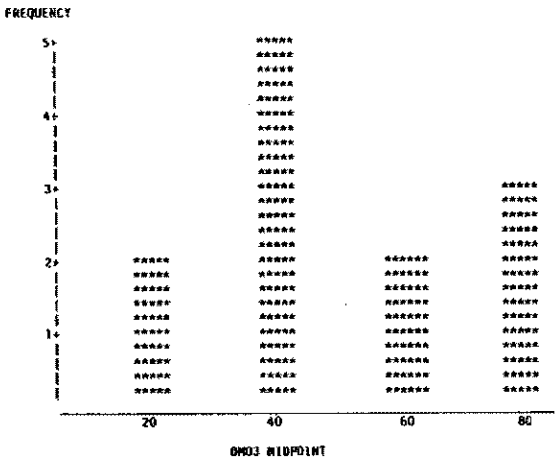
FOR HO: VARIANCES ARE EQUAL, F'= 1.22 WITH 5 AND 5 DF PROB > F'= 0.8327

TTEST OUTPUT FROM THE FIRST SET OF MACRO REPLACEMENT VARIABLES____↑

FIGURE 6 (OUTPUT FROM MACRO EX2)

SAS

FREQUENCY BAR CHART



†THIS BAR CHART WAS DONE BECAUSE THE VALUE OF CODE WAS "C"

FIGURE 9 (OUTPUT FROM MACRO CODCHART)

```
**EXAMPLE 4 USING SYMGET;

  DATA EXAMPS4;
   SET EXAMPS3;

%LET CODE=C
REPORT=SYMGET ('CODE');                    <-THIS SYMGET PASSES THE VALUE OF THE MACRO VARIABLE CODE
IF REPORT='C' THEN GOTO QTR1;                TO THE VARIABLE REPORT

QTR1:

   DATA NULL_;
   SET EXAMPS;
   FILE PRINT NOTITLES HEADER=H;
   QTRSALES=DM01+DM02+DM03;
   IF FIRST OBS THEN LINK H;

   PUT ////
     @1 SUBJECT @10 GROUP @15 DM01 @20 DM02 @25 DM03
     @30 QTRSALES;
     RETURN;

   H:
     PUT @25 'QUARTERLY SALES REPORT';
RETURN;
RETURN;
```

FIGURE 10 (SYMGET PROGRAM STATEMENTS)

QUARTERLY SALES REPORT

1	A	76	87	90	235	<-THE SYMGET USED THE VALUE OF THE VARIABLE
2	B	65	76	82	223	CODE PASSED IT TO THE VARIABLE REPORT, AND
3	B	32	45	12	89	CREATED THIS OUTPUT.
4	B	21	44	52	117	
5	A	43	26	82	151	
6	A	25	36	47	108	
7	B	17	71	37	125	
8	B	52	47	26	125	
9	A	52	54	45	151	
10	B	51	23	61	135	
11	A	68	49	48	165	

FIGURE 11 (OUTPUT FROM THE USE OF THE SYMGET STATEMENT)