

A One-Size-Fits-All Macro to Reduce Repetitive Coding

The SAS macro language is widely used, and often best used, to shorten and simplify, what would otherwise be unwieldy or repetitious program code. This presentation will develop a single macro that will simplify a program that contains a great deal of systematic repetition. A general familiarity with the SAS macro language is presumed, but those unfamiliar with SAS macros may at least gain some understanding of their uses. The project presented here grew out of an actual work assignment at the Research Division of the Texas Comptroller of Public Accounts. Some irrelevant details of that project are omitted.

Among other functions, the Research Division of the state comptroller's agency provides economic data to the public. A request for sales tax data was made by a city official of a very small town. His town did not levy the local option sales tax, and he wanted to estimate the amount that could be obtained through such a levy. He provided a list of taxpayer numbers of businesses both in and near the town and requested from us a listing of aggregate reported total sales by calendar quarter for both the in-city businesses and out-of-city businesses. The request covered a recent seven-year period.

Example 1 presents a solution to the problem with no macro programming. Each input file is a SAS data set with data for one calendar quarter. In the example we extract the quarterly data, selecting on taxpayer number in a series of DATA steps, one step for each quarter from 1987, first quarter, through 1993, fourth quarter. In the example the ellipses represent all the intervening DATA steps. In each step we include the gross sales (GRSAL___) and amount subject to sales tax (AST___) from the input data set and assign an outlet variable (OUT___) with a value of one. This will be used to add up the number of outlets reporting for each quarter.

After this initial series of 28 DATA steps, the resulting data sets (SLS___) are combined in another step. Here the INCITY variable is introduced. For in-city taxpayers its value is "Y"; for out-of-city taxpayers, "N." Now PROC SUMMARY is used to find the total gross sales, amount subject to tax, and number of outlets for all quarters. The totals are separate for in-city and out-of-city taxpayers. The final DATA step produces a permanent output file. In this step we check for amounts that must be omitted to comply with state confidentiality laws. The rule is that if there are fewer than four outlets or taxpayers reporting in a given quarter, the gross sales and amount subject to tax must be kept confidential. Therefore there is a series of statements to check the outlet counts and set corresponding quarterly values to missing when the number of outlets is less than four. Finally, in the same step, a LABEL statement creates labels for all variables. In the example, ellipses are used again to represent repetitive code segments omitted here. Example 2 is a printout of the output data set.

This is all hugely repetitive, so Example 3 presents a version that uses the SAS macro language to make the program more compact. The first macro, DATAQTR, performs a loop cycling the macro variable QTR from 871 through 934, reading the quarterly input data sets and producing corresponding output data sets. Note the %IF statement. This does the arithmetic to skip from, say, 874 to 881, since there are no quarters 875 through 880.

The next step collects all the initial data into one data set as in Example 1, but it invokes a macro, DSQTR, to list all the input data sets. This macro contains a loop structure like the first macro, DATAQTR. The PROC SUMMARY step is the same as in Example 1. The final step contains two macros, one to set confidential data to missing and the other to write appropriate quarterly labels on variables. The important thing to note with these macros is that they are both based on the same looping structure from starting quarter to ending quarter that occurred in the previous macros.

A variant of Example 3 is to create two global macro variables, START and END, which are set to 871 and 934 and used as the beginning and end values for each of the macros (Example 4). Now, if we want to run the program for a different time period, we can do so by changing the values at the beginning of the program.

Examples 3 and 4 present a reasonably satisfactory solution, but it will be noticed that all the macros in them are based on the same structure, a loop using quarters as values. Why should not these be reduced to a single, simple macro that does the looping and receives the varying SAS program code as a parameter?

Example 5 is not a functioning program. It is just a sketch of what is suggested here. The macro QUARTER accepts a parameter called SASCODE, which is whatever repetitive SAS program code is needed, and reproduces the SAS code for all desired time periods. SASCODE is simply plugged into the loop. Whenever the local macro variable QTR occurs in the SAS code, the appropriate quarter number will be substituted by the incrementing %DO statement. Thus, the first group of DATA steps to input the sales data is performed by invoking the macro QUARTER and sending all the program code for one DATA step as a parameter.

This, however, will not work. The problem is with the macro variable QTR, which occurs in the SAS code being sent as a parameter. In the macro invocation the macro processor attempts to resolve the value of QTR in the parameter. In other words, it is searching for a global macro variable QTR and not finding it. Since that variable is a local variable within the macro, it is undefined at the time of invocation. Of course, we do not want the variable to be resolved on invocation; we want it sent as is into the loop in the macro, within which it is a defined local macro variable.

The solution is in Example 6. The %NRSTR (no rescan string) macro function is called with the SAS code as the argument. The %NRSTR function prevents the macro processor from attempting to resolve any macro variable inside the argument string. In

other words, the function instructs the macro processor to ignore the usual meaning of the ampersand. Thus, the %NRSTR function allows us to invoke the macro and place the code, including the macro variable QTR, ampersand and all, inside the loop. At this point the opposite problem occurs. Now that the code has been placed in the loop and the local variable QTR has been defined, the macro processor must be told to interpret the macro variable as it normally would. This is done with the %UNQUOTE function, which reverses the effect of the %NRSTR function and causes the macro processor to resolve QTR as the indicated value within the loop.

The use of this macro drastically shortens the program, reducing it to a single page. Although it may look challenging at first, it reduces the cluttered appearance of highly repetitious SAS program code or of several macros interspersed with conventional SAS code segments. It can easily be saved and reused in other programs where similar repetition occurs, and, of course, it can be adapted to any kind of program code that combines exact repetition with uniformly incrementing or decrementing elements. This macro can be changed instantly and with little chance for error whenever one wants to run the program for a different time period. Even a user with no SAS programming knowledge can be taught to make desired changes of time periods and run the job. Finally, this may serve as a small hint of the SAS macro language's ability to turn ad hoc programs into highly flexible and sometimes complex information systems.

Example 1: No Macros

```
LIBNAME IN 'USER.R30.SAS.SIXQUART.DATAPAY.SIC87';
LIBNAME OUT 'USER.R30.MLIN460.SAS.SALES.DATA8793' UNIT=3380
VOLSER=USER38 SPACE=(TRK,(1,1),RLSE) DISP=(NEW,CATLG,DELETE) ;
```

```
DATA SLS871;
  SET IN.DATA871 (KEEP=TAXNO GRSAL871 AST871);
  WHERE TAXNO IN (30114150433, 24504856089, 24664186350,
    24539600718, 30008189943, 30116073112, 24636841686, 24298089053,
    24376274197, 24502456213, 24537467094, 24627606973, 24647052307,
    25234248430, 24565820156, 24804404127, 24565205739, 22274405400,
    24443811252, 24648411734, 24845280585, 24673092573, 24526292511,
    25258008751, 24542608252, 24606003721, 24534183330);
  LENGTH OUT871 2;
  OUT871 = 1;
```

. . .

```
DATA SLS934;
  SET IN.DATA934 (KEEP=TAXNO GRSAL934 AST934);
  WHERE TAXNO IN (30114150433, 24504856089, 24664186350,
    24539600718, 30008189943, 30116073112, 24636841686, 24298089053,
    24376274197, 24502456213, 24537467094, 24627606973, 24647052307,
    25234248430, 24565820156, 24804404127, 24565205739, 22274405400,
    24443811252, 24648411734, 24845280585, 24673092573, 24526292511,
    25258008751, 24542608252, 24606003721, 24534183330);
  LENGTH OUT934 2;
  OUT934 = 1;
```

```
DATA SLSHIST (DROP=TAXNO);
  SET SLS871 SLS872 SLS873 SLS874 SLS881 SLS882 SLS883 SLS884
    SLS891 SLS892 SLS893 SLS894 SLS901 SLS902 SLS903 SLS904
    SLS911 SLS912 SLS913 SLS914 SLS921 SLS922 SLS923 SLS924
    SLS931 SLS932 SLS933 SLS934;
  LENGTH INCITY $1;
  IF TAXNO IN (30114150433, 24504856089, 24664186350, 24539600718,
    30008189943, 30116073112, 24636841686, 24298089053, 24376274197,
    24502456213, 24537467094) THEN INCITY = 'N';
  ELSE INCITY = 'Y';
```

```
PROC SUMMARY NWAY;
  CLASS INCITY;
  VAR GRSAL: AST: OUT: ;
  OUTPUT OUT=SLSHIST (DROP=_TYPE_ _FREQ_) SUM=;
```

```
DATA OUT.SLSHIST;
  SET SLSHIST;
  IF OUT871 < 4 THEN
    DO;
      GRSAL871 = .;
      AST871 = .;
    END;
```

IF OUT934 < 4 THEN

DO;

GRSAL934 = .;

AST934 = .;

END;

LABEL GRSAL871='SALES*871'

AST871 = 'AMT SUBJ*TO TAX 871'

OUT871 = 'OUTLETS*871'

GRSAL934='SALES*934'

AST934 = 'AMT SUBJ*TO TAX 934'

OUT934 = 'OUTLETS*934' ;

Example 2

INCITY	SALES 871	SALES 872	SALES 873	SALES 874	SALES 881	SALES 882	SALES 883	SALES 884	SALES 891
N	151427	232312	236513	165768	174006	216758	232106	164042	169664
Y	.	.	.	55072	259020	472912	436912	179994	.
SALES 892	SALES 893	SALES 894	SALES 901	SALES 902	SALES 903	SALES 904	SALES 911	SALES 912	SALES 913
229670	239027	182637	185537	243110	271085	250784	206431	281837	308130
.	.	278778	522694	658572	452064	288415	414190	679737	508704
SALES 914	SALES 921	SALES 922	SALES 923	SALES 924	SALES 931	SALES 932	SALES 933	SALES 934	
298139	173328	229002	242127	217763	169205	237866	244303	378261	
253750	220530	353271	350959	315103	259482	416799	569586	602038	
AMT SUBJ TO TAX 871	AMT SUBJ TO TAX 872	AMT SUBJ TO TAX 873	AMT SUBJ TO TAX 874	AMT SUBJ TO TAX 881	AMT SUBJ TO TAX 882				
65033	97762	95979	77617	65025	86196				
.	.	.	41330	212871	400822				
AMT SUBJ TO TAX 883	AMT SUBJ TO TAX 884	AMT SUBJ TO TAX 891	AMT SUBJ TO TAX 892	AMT SUBJ TO TAX 893	AMT SUBJ TO TAX 894				
82743	57087	61243	87507	92700	71058				
353236	96915	.	.	.	184218				
AMT SUBJ TO TAX 901	AMT SUBJ TO TAX 902	AMT SUBJ TO TAX 903	AMT SUBJ TO TAX 904	AMT SUBJ TO TAX 911	AMT SUBJ TO TAX 912				
71973	95580	89737	70941	75280	95111				
439528	547476	322830	160162	320983	540094				
AMT SUBJ TO TAX 913	AMT SUBJ TO TAX 914	AMT SUBJ TO TAX 921	AMT SUBJ TO TAX 922	AMT SUBJ TO TAX 923	AMT SUBJ TO TAX 924				
90031	97452	70611	94214	95411	136689				
374377	143297	124314	202479	187410	148562				
AMT SUBJ TO TAX 931	AMT SUBJ TO TAX 932	AMT SUBJ TO TAX 933	AMT SUBJ TO TAX 934	OUTLETS 871	OUTLETS 872	OUTLETS 873			
70444	101610	93138	133236	6	6	5			
111836	204163	273389	259042	3	3	3			
OUTLETS 874	OUTLETS 881	OUTLETS 882	OUTLETS 883	OUTLETS 884	OUTLETS 891	OUTLETS 892	OUTLETS 893	OUTLETS 894	
5	5	6	5	7	6	7	6	8	
8	5	5	5	9	2	3	3	12	
OUTLETS 901	OUTLETS 902	OUTLETS 903	OUTLETS 904	OUTLETS 911	OUTLETS 912	OUTLETS 913	OUTLETS 914	OUTLETS 921	
5	5	5	9	6	7	6	9	5	
4	4	5	12	6	6	6	13	6	
OUTLETS 922	OUTLETS 923	OUTLETS 924	OUTLETS 931	OUTLETS 932	OUTLETS 933	OUTLETS 934			
6	4	9	4	5	4	7			
7	7	15	7	7	9	18			

Example 3: Several Macros

```

LIBNAME IN 'USER.R30.SAS.SIXQUART.DATAPAY.SIC87';
LIBNAME OUT 'USER.R30.MLIN460.SAS.SALES.DATA8793' UNIT=3380
VOLSER=USER38 SPACE=(TRK,(1,1),RLSE) DISP=(NEW,CATLG,DELETE) ;

%MACRO DATAQTR;  /* CREATE DATASETS SLS871, SLS872, . . . SLS934;
%LOCAL QTR;
%DO QTR = 871 %TO 934;
  %IF %SUBSTR(&QTR,3) = 5 %THEN %LET QTR = %EVAL(&QTR+6);
  DATA SLS&QTR;
    SET IN.DATA&QTR (KEEP=TAXNO GRSAL&QTR AST&QTR);
    WHERE TAXNO IN (30114150433, 24504856089, 24664186350,
      24539600718, 30008189943, 30116073112, 24636841686, 24298089053,
      24376274197, 24502456213, 24537467094, 24627606973, 24647052307,
      25234248430, 24565820156, 24804404127, 24565205739, 22274405400,
      24443811252, 24648411734, 24845280585, 24673092573, 24526292511,
      25258008751, 24542608252, 24606003721, 24534183330);
    LENGTH OUT&QTR 2;
    OUT&QTR = 1;
  %END;
%MEND DATAQTR;

%DATAQTR

%MACRO DSQTR;  /* LIST DATASETS SLS871, SLS872, . . . SLS934;
%LOCAL QTR;
%DO QTR = 871 %TO 934;
  %IF %EVAL(&QTR-&QTR/10*10) > 4 %THEN %LET QTR = %EVAL(&QTR+6);
  SLS&QTR
%END;
%MEND DSQTR;

DATA SLSHIST (DROP=TAXNO);
SET %DSQTR ;
LENGTH INCITY $1;
IF TAXNO IN (30114150433, 24504856089, 24664186350, 24539600718,
  30008189943, 30116073112, 24636841686, 24298089053, 24376274197,
  24502456213, 24537467094) THEN INCITY = 'N';
ELSE INCITY = 'Y';

PROC SUMMARY NWAY;
CLASS INCITY;
VAR GRSAL: AST: OUT: ;
OUTPUT OUT=SLSHIST (DROP=_TYPE_ _FREQ_) SUM=;

%MACRO MISSQTR;  /* IF OUT___ < 4, SET GRSAL___ AND AST___ TO MISSING ;
%LOCAL QTR;
%DO QTR = 871 %TO 934;
  %IF %SUBSTR(&QTR,3) = 5 %THEN %LET QTR = %EVAL(&QTR+6);
  IF OUT&QTR < 4 THEN
    DO;
      GRSAL&QTR = .;

```

```

        AST&QTR = .;
    END;
%END;
%MEND MISSQTR;

%MACRO LABELQTR;  %* SET LABELS FOR GRSAL____, AST____, OUT____ ;
    %LOCAL QTR;
    %DO QTR = 871 %TO 934;
        %IF %EVAL(&QTR-&QTR/10*10) > 4 %THEN %LET QTR = %EVAL(&QTR+6);
        GRSAL&QTR="SALES*&QTR"
        AST&QTR  ="AMT SUBJ*TO TAX &QTR"
        OUT&QTR  ="OUTLETS*&QTR"
    %END;
%MEND LABELQTR;

DATA OUT.SLSHIST;
    SET SLSHIST;
    %MISSQTR

    LABEL %LABELQTR ;

```

Example 4: Global Variables for Starting and Ending Quarters

```

LIBNAME IN 'USER.R30.SAS.SIXQUART.DATAPAY.SIC87';
LIBNAME OUT 'USER.R30.MLIN460.SAS.SALES.DATA8793' UNIT=3380
  VOLSER=USER38 SPACE=(TRK,(1,1),RLSE) DISP=(NEW,CATLG,DELETE) ;

%LET START = 871; /* THESE TWO VARIABLES CAN */
%LET END   = 934; /* BE CHANGED EVERY QUARTER */

%MACRO DATAQTR; /* CREATE DATASETS SLS871, SLS872, . . . SLS934;
  %LOCAL QTR;
  %DO QTR = &START %TO &END;
    %IF %SUBSTR(&QTR,3) = 5 %THEN %LET QTR = %EVAL(&QTR+6);
    DATA SLS&QTR;
      SET IN.DATA&QTR (KEEP=TAXNO GRSAL&QTR AST&QTR);
      WHERE TAXNO IN (30114150433, 24504856089, 24664186350,
        24539600718, 30008189943, 30116073112, 24636841686, 24298089053,
        24376274197, 24502456213, 24537467094, 24627606973, 24647052307,
        25234248430, 24565820156, 24804404127, 24565205739, 22274405400,
        24443811252, 24648411734, 24845280585, 24673092573, 24526292511,
        25258008751, 24542608252, 24606003721, 24534183330);
      LENGTH OUT&QTR 2;
      OUT&QTR = 1;
    %END;
  %MEND DATAQTR;

%DATAQTR

%MACRO DSQTR; /* LIST DATASETS SLS871, SLS872, . . . SLS934;
  %LOCAL QTR;
  %DO QTR = &START %TO &END;
    %IF %EVAL(&QTR-&QTR/10*10) > 4 %THEN %LET QTR = %EVAL(&QTR+6);
    SLS&QTR
  %END;
%MEND DSQTR;

DATA SLSHIST (DROP=TAXNO);
  SET %DSQTR ;
  LENGTH INCITY $1;
  IF TAXNO IN (30114150433, 24504856089, 24664186350, 24539600718,
    30008189943, 30116073112, 24636841686, 24298089053, 24376274197,
    24502456213, 24537467094) THEN INCITY = 'N';
  ELSE INCITY = 'Y';

PROC SUMMARY NWAY;
  CLASS INCITY;
  VAR GRSAL: AST: OUT: ;
  OUTPUT OUT=SLSHIST (DROP=_TYPE_ _FREQ_) SUM=;

%MACRO MISSQTR; /* IF OUT___ < 4, SET GRSAL___ AND AST___ TO MISSING ;
  %LOCAL QTR;
  %DO QTR = &START %TO &END;
    %IF %EVAL(&QTR-&QTR/10*10) > 4 %THEN %LET QTR = %EVAL(&QTR+6);

```

```

IF OUT&QTR < 4 THEN
  DO;
    GRSAL&QTR = .;
    AST&QTR = .;
  END;
%END;
%MEND MISSQTR;

%MACRO LABELQTR;  %* SET LABELS FOR GRSAL____, AST____, OUT____ ;
  %LOCAL QTR;
  %DO QTR = &START %TO &END;
    %IF %EVAL(&QTR-&QTR/10*10) > 4 %THEN %LET QTR = %EVAL(&QTR+6);
    GRSAL&QTR="SALES*&QTR"
    AST&QTR  ="AMT SUBJ*TO TAX &QTR"
    OUT&QTR  ="OUTLETS*&QTR"
  %END;
%MEND LABELQTR;

DATA OUT.SLSHIST;
  SET SLSHIST;
  %MISSQTR

  LABEL %LABELQTR ;

```

Example 5: Sketch for a Single Macro--Not a Workable Program

```
LIBNAME IN 'USER.R30.SAS.SIXQUART.DATAPAY.SIC87';
LIBNAME OUT 'USER.R30.MLIN460.SAS.SALES.DATA8793' UNIT=3380
VOLSER=USER38 SPACE=(TRK,(1,1),RLSE) DISP=(NEW,CATLG,DELETE) ;
```

```
%MACRO QUARTER(SASCODE);
  %LOCAL QTR;
  %DO QTR = 871 %TO 934;
    %IF %SUBSTR(&QTR,3) = 5 %THEN %LET QTR = %EVAL(&QTR+6);
    &SASCODE
  %END;
%MEND QUARTER;
```

```
%QUARTER(
  DATA SLS&QTR;
    SET IN.DATA&QTR (KEEP=TAXNO GRSAL&QTR AST&QTR);
    WHERE TAXNO IN (30114150433, 24504856089, 24664186350,
      24539600718, 30008189943, 30116073112, 24636841686, 24298089053,
      24376274197, 24502456213, 24537467094, 24627606973, 24647052307,
      25234248430, 24565820156, 24804404127, 24565205739, 22274405400,
      24443811252, 24648411734, 24845280585, 24673092573, 24526292511,
      25258008751, 24542608252, 24606003721, 24534183330);
    LENGTH OUT&QTR 2;
    OUT&QTR = 1; )
```

```
DATA SLSHIST (DROP=TAXNO);
  SET %QUARTER(SLS&QTR) ;
  LENGTH INCITY $1;
  IF TAXNO IN (30114150433, 24504856089, 24664186350, 24539600718,
    30008189943, 30116073112, 24636841686, 24298089053, 24376274197,
    24502456213, 24537467094) THEN INCITY = 'N';
  ELSE INCITY = 'Y';
```

```
PROC SUMMARY NWAY;
  CLASS INCITY;
  VAR GRSAL: AST: OUT: ;
  OUTPUT OUT=SLSHIST (DROP=_TYPE_ _FREQ_) SUM=;
```

```
DATA OUT.SLSHIST;
  SET SLSHIST;
  %QUARTER(
    IF OUT&QTR < 4 THEN
      DO;
        GRSAL&QTR = .;
        AST&QTR = .;
      END; )
```

```
LABEL %QUARTER(
  GRSAL&QTR="SALES*&QTR"
  AST&QTR  ="AMT SUBJ*TO TAX &QTR"
  OUT&QTR  ="OUTLETS*&QTR" ) ;
```

Example 6: Single Macro

```
LIBNAME IN 'USER.R30.SAS.SIXQUART.DATAPAY.SIC87';
LIBNAME OUT 'USER.R30.MLIN460.SAS.SALES.DATA8793' UNIT=3380
VOLSER=USER38 SPACE=(TRK,(1,1),RLSE) DISP=(NEW,CATLG,DELETE) ;

%MACRO QUARTER(SASCODE);
  %LOCAL QTR;
  %DO QTR = 871 %TO 934;
    %IF %SUBSTR(&QTR,3) = 5 %THEN %LET QTR = %EVAL(&QTR+6);
    %UNQUOTE(&SASCODE)
  %END;
%MEND QUARTER;

%QUARTER(%NRSTR(
  DATA SLS&QTR;
    SET IN.DATA&QTR (KEEP=TAXNO GRSAL&QTR AST&QTR);
    WHERE TAXNO IN (30114150433, 24504856089, 24664186350,
      24539600718, 30008189943, 30116073112, 24636841686, 24298089053,
      24376274197, 24502456213, 24537467094, 24627606973, 24647052307,
      25234248430, 24565820156, 24804404127, 24565205739, 22274405400,
      24443811252, 24648411734, 24845280585, 24673092573, 24526292511,
      25258008751, 24542608252, 24606003721, 24534183330);
    LENGTH OUT&QTR 2;
    OUT&QTR = 1; ))

DATA SLSHIST (DROP=TAXNO);
  SET %QUARTER(%NRSTR(SLS&QTR)) ;
  LENGTH INCITY $1;
  IF TAXNO IN (30114150433, 24504856089, 24664186350, 24539600718,
    30008189943, 30116073112, 24636841686, 24298089053, 24376274197,
    24502456213, 24537467094) THEN INCITY = 'N';
  ELSE INCITY = 'Y';

PROC SUMMARY NWAY;
  CLASS INCITY;
  VAR GRSAL: AST: OUT: ;
  OUTPUT OUT=SLSHIST (DROP=_TYPE_ _FREQ_) SUM=;

DATA OUT.SLSHIST;
  SET SLSHIST;
  %QUARTER(%NRSTR(
    IF OUT&QTR < 4 THEN
      DO;
        GRSAL&QTR = .;
        AST&QTR = .;
      END;
  ))

LABEL %QUARTER(%NRSTR(
  GRSAL&QTR="SALES*&QTR"
  AST&QTR  ="AMT SUBJ*TO TAX &QTR"
  OUT&QTR  ="OUTLETS*&QTR"  )) ;
```