Using CALL SYMPUT to Generate Dynamic Columns in Reports

Suwen Li, Sai Ma, Regan Li, Bob Lan

INTRODUCTION

When creating reports, we often want to make the report respond dynamically to data. If the headers and the number of columns in the report are unknown, it is helpful when they change dynamically depending on the data. As a powerful SAS® procedure, PROC TABULATE can produce dynamic results in most cases. This poster describes how to use the CALL SYMPUT routine and PROC REPORT to generate dynamic columns in reports in cases where PROC TABULATE does not yield the desired results.

CALL SYMPUT Routine

The Syntax for this routine:
CALL SYMPUT (macro-variable, value)
The first argument can have 3 forms:

- A character string enclosed in quotation marks, for example, *call symput('new','testing')*
- The name of a character variable in a DATA step, for example, *call symput(position,player)*
- A character expression that produces a series of macro variables, for example, *call symput ('POS'||left(_n_), position)*

PROC TABULATE

Basic Example

Illustrate the advantage of PROC TABULATE in generating reports automatically adjusting to data.

```
proc tabulate data=popl;
  class treatc;
  var age;
  table age*(mean std), treatc;
run;
```

SAS Output

		TREATC		
		ABC	DEF	PLACEBO
Age	Mean	37.11	37.31	36.8
	Std	12.41	12.12	13.12

The column headers are from the values of the variable TREATC defined in the CLASS statement. When the values of TREATC are changed, the column headers and number in SAS output will change automatically to reflect the change to treatment group in new data.

MORE COMPLEX REPORTS WITH DYNAMIC COLUMNS

Demographic Summary Table by Treatment

Because the report layout is complicated, it needs a few steps of calculations. Thus, PROC TABULATE cannot deal with this type of summary table. Instead, PROC MEANS, PROC FREQ and PROC REPORT are usually used together to create it. After simple calculation, the statistics of age and frequency distribution of gender are printed in below.

CLA	CLAC	TREAT	TREATC	_MEANC	_STDC	_MEDIANC	_MINMAXC	_NC
1	Age (yrs)	0	Placebo	36.8	13.1	37.8	21, 52	14
1	Age (yrs)	1	ABC	37.1	12.4	35.0	20, 58	14
1	Age (yrs)	2	DEF	37.3	12.1	37.0	20, 58	20

CLA	CLAC	SUBCLA	SUBCLAC	TREAT	TREATC	COUNTC
3	Sex, n(%)	12	Male	0	Placebo	3 (75.0)
3	Sex, n(%)	12	Male	1	ABC	7 (50.0)
3	Sex, n(%)	12	Male	2	DEF	8 (40.0)
3	Sex, n(%)	13	Female	0	Placebo	1 (25.0)
3	Sex, n(%)	13	Female	1	ABC	7 (50.0)
3	Sex, n(%)	13	Female	2	DEF	12 (60.0)

After data transposed:

CLA	CLAC	_NAME_	_0	_1	_2
1	Age (yrs)	_MEANC	36.8	37.1	37.3
1	Age (yrs)	_STDC	13.1	12.4	12.1
1	Age (yrs)	_MEDIANC	37.8	35.0	37.0
1	Age (yrs)	_MINMAXC	21, 52	20, 58	20, 58
1	Age (yrs)	_NC	4	14	20

CLA	CLAC	SUBCLA	SUBCLAC	_0	_1	_2
3	Sex, n(%)	12	Male	3 (75.0)	7 (50.0)	8 (40.0)
3	Sex, n(%)	13	Female	1 (25.0)	7 (50.0)	12 (60.0)

Using PROC REPORT to generate output

```
proc sort data=out3 out=treat nodupkey;
  by treat;
run;

data _null_;
 set treat end=last;
 call symput('Treat'||left(put(_n_,best.)),
left(put(treat,best.)));
 call symput('Treatc'||left(put(_n_,best.)), treatc );
 if last then do;
     call symput('total', left(put(_n_,best.)));
  end;
run;

%macro report;
%if &total=2 %then %do;
  %let cwidth=39;
%end;

%if &total=3 %then %do;
  %let cwidth=26;
%end;
```

```
proc report data=report2 nowd missing headline headskip spacing=0
split='@' ;
  column ('__' cla clac subcla
        %do i=1 %to &total;
          _&&treat&i
        %end;
        );
   define cla      /group order=internal noprint;
   define clac     /group order noprint;
   define subcla /order=internal width=54 ' '  flow left f=subcla.
spacing=0;

   %do j=1 %to &total;
     define _&&treat&j  / width=&cwidth   flow
"&&treatc&j@(N=%left(&&t&j))";
   %end;

   break after clac / skip;
   compute before clac;
    line @1 clac $ left;
   endcomp;
run;

%mend;
%report;
```

SAS Output

	Placebo (N=4)	ABC (N=14)	DEF (N=20)
Age (yrs)			
Mean	36.8	37.1	37.3
SD	13.1	12.4	12.1
Median	37.8	35.0	37.0
Min, Max	21, 52	20, 58	20, 58
n	4	14	20
Sex, n(%)			
Male	3 (75.0)	7 (50.0)	8 (40.0)
Female	1 (25.0)	7 (50.0)	12 (60.0)

Clinical Laboratory Results Listing

- In the output, test dates and visit per subject ID are requested to be presented in column headings.
- Different subject IDs may have different test dates and some subjects may miss a few visits.
- The number of columns will change by subjects and pages.
- The column headings will change by subjects and pages also.
- The number and headings of columns for each subject is fixed. It may change when the dataset is updated.

For example, one subject may have only 5 columns, and one subject may have as many as 23 columns. Thus, the report will have only 5 columns in some page, but some may have columns more or less than 5.

Using CALL SYMPUT to Generate Dynamic Columns in Reports

ADLB Printout

SUBJID	PARMGRP	PARAM	ADT	LBSTRESU	LBSTRESC
001	CHEMISTRY	Protein, Total	2011-08-11	g/L	77.0
001	CHEMISTRY	Bilirubin, Total	2011-08-11	umol/L	1.81
001	CHEMISTRY	Triglycerides	2011-08-11	mmol/L	2.21

SAS Code

Only keep unique visit date for each lab test category and subject in dataset LBDT.

```
proc sort data=adlb out=lbdt(keep=subjid parmgrp adt) nodupkey;
  by subjid parmgrp adt;
run;
```

- Assign each subject's visit date to macro variables.
- Create macro variables for each subject's maximum number of visits.
- Create a macro variable for total number of lab test groups for all subjects.

```
  data lbdt2;
    set lbdt end=last;
    by subjid parmgrp adt;
    retain parmgrp2 subjid2;
    if first.parmgrp then seq=.;
    seq+1;

    if _n_=1 then do;
       grpseq=1;
       parmgrp2=parmgrp;
       subjid2=subjid;
    end;
    if parmgrp2^=parmgrp or subjid2^=subjid then do;
       grpseq+1;
       parmgrp2=parmgrp;
       subjid2=subjid;
    end;
    call symput('dt'||compress(put(grpseq,best.)||put(seq,best.)),
    put(adt,yymmdd10.));

    if last.parmgrp then do;
       call symput('tot'||compress(put(grpseq,best.)), put(seq,best.));
    end;
    if last then do;
       call symput('lastgrp', put(grpseq,best.));
    end;
    drop parmgrp2 subjid2;
  run;
```

Merge the sequence number back to original lab test dataset.

```
  data adlb2;
    merge adlb lbdt2;
    by subjid parmgrp adt;
    param='  '||trim(left(param));
  run;
```

Transpose the data and specify the sequence number which represents each visit date as variable names in the output dataset.

```
  proc sort data=adlb2;
    by subjid parmgrp grpseq param lbstresu;
  run;

  proc transpose data=adlb2 out=adlb3;
    by subjid parmgrp grpseq param lbstresu;
    id seq;
    var lbstresc;
  run;
```

Generate output.

Generate lab test listings

```
%macro lb;
%do i=1 %to &lastgrp;
proc report data=adlb3 nowd headskip headline split='!';
  where grpseq=&i;
  column ('__' subjid parmgrp param lbstresu
    %do j=1 %to &&tot&i;
       _&j
    %end;
    );
  define subjid/group noprint;
  define parmgrp/group noprint;
  define param/group order width=36 flow left id  "Clinical
Laboratory Group!   Parameter" spacing=0;
  define lbstresu/group order width=08 flow left id  'Unit' spacing=0;
  %do j=1 %to &&tot&i;
     define _&j/"&&dt&i&j" width=20;
   %end;
  compute before parmgrp ;
    length c0d $100;
    c0d=trim(left(parmgrp));
     line @01 c0d  $100.;
  endcomp;

  compute before _page_;
     line 132*'_';
     line @1 'Patient ID:' @13 subjid $132.;
   endcomp;
 run;
%end;
%mend;
%lb;
```

SAS Output

```
Patient ID: 001

Clinical Laboratory Group
  Parameter                          Unit   2011-08-11 2011-09-12
                                            (Visit 2)  (Visit 3)

CHEMISTRY
  Alanine Aminotransferase (SGPT)    U/L        25         32
  Albumin                            g/L        48         46
  Alkaline Phosphatase               U/L        74         73
  .
  .
  .

Patient ID: 002

Clinical Laboratory Group
  Parameter                          Unit   2011-07-21 2011-08-12 2011-10-12
                                            (Visit 1)  (Visit 2)  (Visit 4)

CHEMISTRY
  Alanine Aminotransferase (SGPT)    U/L        25         32         47
  Albumin                            g/L        48         46         50
  Alkaline Phosphatase               U/L        74         73         80
```

CONCLUSION

An advantage of PROC REPORT is its ability to define the columns and present results in a report after data manipulation when producing complex reports. Combined with the use of CALL SYMPUT, it can create dynamic columns depending on data in those complex reports which PROC TABLUATE cannot handle.