Paper 32-25

One Macro Does It All – Advanced Enhancement of PROC REPORT

Peng-fang Yen, ASG, Vernon Hills, IL and Jeff Gudmundson, TAP HOLDINGS, Deerfield, IL

ABSTRACT

PROC REPORT is the most powerful and productive tool among all Base SAS® report generators. One of its most appreciated things is that there is no need to pre-figure the column position for each displayed variable and its header. However, some desired reporting-formats can not be accomplished by this procedure such as page break at a value of variable when its whole group cannot fit on a page, or wrapping up values of an ACROSS variable on the next page. And, it is inconvenient and time-consuming for coders to use other SAS report generators for these desired formats. A macro to accomplish all of these with the full use of PROC REPORT has been developed at TAP Holdings Inc.. Additionally, it handles content-dependent footnotes, places page-numbering at a fixed position at the bottom of page and allows a combined use of aligning title and footnote texts on the left, on the right and in the center of page. This paper will describe the macro's parameters, settings and functions, as well as demonstrate its use. It is targeted at those who are looking for an efficient way of generating reports or those who have experience with PROC REPORT. The macro is developed using SAS version 6.12 on the NT operating system. It can be used in other operating systems.

INTRODUCTION

This macro grew out of standardizing the customized reports at TAP Holdings Inc., a pharmaceutical company. In order to make reports more readable and understandable internally and externally, TAP wanted to have reports in standard and customized format across all projects. However, customizing reports required a great amount of SAS® coding work. This kind of work would become a burden to SAS coders when there was a deadline to meet.

PROC REPORT was evaluated to be the most powerful report-generated procedure among all SAS report generators. It provides much more flexibility than other generators such as transposing values of a variable into columns, flowing a text value of a variable to the next row, providing compute block, break and rbreak statements to facilitate the creation of sophisticated reports, etc. However, some preferred outcomes are difficult to produce such as wrapping values of an ACROSS variable on the next page with appropriate pagination, moving a group of value of variable on the next page, and placing the customized pagination at a fixed position.

Generally, PROC REPORT was the device that TAP was looking for, but needed an enhancement to accomplish TAP's expectation, the standardization of customized reports. A macro, TABRPT, was proposed to enhance the PROC REPORT procedure. Its primary functions are to control page breaks as wished, wrap up ACROSS values on the next consecutive page, display footnotes conditional on the contents of a page, and provide a flexible use of titles and footnotes. The flexible use means a capacity of stating as many titles and footnotes as the SAS system allows via three macro parameters and aligning their texts on the left, on the right and in the center of page. Moreover, it displays the page numbering in format of "Page x of y" at the right bottom corner of page and breaks up sections (title, header, body and footnote) of page with three solid lines.

This paper will describe the TABRPT macro parameters, demonstrate its use, and illustrate methods used for wrapping all ACROSS values on consecutive pages, justifying the alignment of title and footnote texts, controlling page breaks as defined, and generating a report.

MACRO PARAMETERS OF TABRPT

The macro parameters used in this TABRPT macro are defined as DATA, SORTORD, COLUMNS, DEFINE, PANLNO, TITLE, FOOTNOTE and CONDFT. The DATA parameter passes a SAS data set to generate a report. SORTORD specifies a list of variables in an order of sorting the contents of report. When there are variables defined as ACROSS type, this list should include all of the variables in the COLUMN statement before the first ACROSS variable and all the ACROSS variables. SORTORD also supports an option of indicating a BREAK variable by adding the + symbol preceding one of the variables in the list. This variable cannot be an ACROSS variable. The BREAK variable signifies a page break when its value changes and the whole group of this new value cannot be displayed on a page. The COLUMNS parameter functions exactly like the COLUMN syntax supported by the PROC REPORT procedure. The DEFINE parameter is a place to state DEFINE, BREAK, RBREAK, COMPUTE BLOCK, LINE, and BY statements. PANLNO indicates the maximum number of different values of the first ACROSS variable that can be displayed on a page. The default of PANLNO is 2. The TITLE parameter includes all the title statements to be printed on a report. FOOTNOTE performs exactly like the TITLE parameter. CONDFT consists of footnotes displayed depending on the contents of the page. Title and footnote texts can be left-justified, right-justified or centered by setting j equal to l, r, or c, respectively. These parameters were set up with an assumption that reports will be generated in a nowindowing, nocenter, missing-value-displayed environment with a split character being ^.

USAGE OF TABRPT

Below is an example of calling this TABRPT macro:

```
%TABRPT(DATA   =statemp
       ,SORTORD=analyc +trt dummy ptno prd schtmc
       ,PANLNO =10
       ,COLUMNS=%str(analyc trt dummy ptno prd schtmc,(charrslt condft1))
       ,DEFINE =%str(define analyc  /group noprint;
                     define trt     /group noprint;
                     define dummy   /group odrer=internal noprint;
                     define schtmc  /across order=data format=gtform. "" center;
                     define ptno    /group order=data format=subform. width=10 "Subject" center;
                     define prd     /group width=7 "Period" CENTER;
                     define charrslt/display width=8 spacing=2 "" center nozero;
                     define condft1 /display format=cf. width=1 spacing=0 "" center nozero;
                     compute before analyc;
                        line @3 "ANALYTE:  " analyc analyfmt.;
                     endcomp;
                     compute before trt;
```

```
                    line @3 'REGIMEN:  ' trt treat.;
                    line "";
                  endcomp;
                  break after analyc/page;
                  break after trt   /skip;
                  break after dummy /skip;
                 )
      ,TITLE  =%str(title1 j=l 'DUMMY DRUG: STUDY 99-999';
                  title2 j=l "%SYSFUNC(TODAY(),TODAY.) <SUGI25.SAS YENP>";
                  title3 j=l 'SAMPLE EXAMPLE REPORT';
                  title5 j=c 'SUGI 25';
                  title7 j=c 'REPORT GENERATED BY TABRPT';
                 )
      ,CONDFT =%str(condft1 j=l "*  indicating value > 2500.")
      )
```

In this calling, the STATEMP data is passed to create a report using its variables -- analyc, trt, dummy, ptno, prd, schtmc, charrslt, and condft1. Using the list in SORTORD, the report will be sorted by analyc first, followed by trt, dummy, ptno, prd and schtmc. It also will display all of the 16 different values in schtmc horizontally onto two pages since PANLNO is 10 and schtmc is ACROSS-type. The first page will have 10 of these 16 while the consecutive page contains the rest of them. In addition to the analyc variable defined to control the page breaks, trt in SORTORD specified as the BREAK variable will signify a new page to start when the whole group of its value cannot fit on a page. For the title and footnote statements, the texts in title1, title2 and title3 contained in the TITLE parameter will be lined up on the left of the 1st, 2nd and 3rd rows of each page, respectively. Those in title5 and title7 will be centered in the 5th and 7th rows. The conditional footnote, condft1 stated in CONDFT, will be displayed right below the third solid line only if the condft1 variable contains a value of 1 for some observations on a page. The CONDFT1 footnote will be justified on the left of the page.

Please see Table 1 on the last page of this paper for the output produced by this code.

WRAPPING ACROSS VALUES

When all of values in an ACROSS variable are unable to be exhibited on a page, the PROC REPORT procedure will split the values on multiple pages. Frequently, these values are not appropriately split. In most cases, a group of an ACROSS value will be separated on two pages. The TABRPT uses PANLNO to control this splitting by grouping all ACROSS values into several sets and assigns the grouping values to a variable named _gpacs. Consider the example where an ACROSS variable contains 16 different values and PANLNO is defined as 10. Then, _gpacs will have a value of 1 associated with the first ten values and 2 for the rest. A page break occurs when _gpacs's value changes. Below is an example of the code of TABRPT for this step using the data described in the section of usage of TABRPT.

```
/** To obtain the different values in an ACROSS variable, SCHMC.    **/

proc sort data=statemp out=_acsvar nodupkey;
     by schtmc;
run;

/** To create an variable, _GPACS, grouping the ACROSS values by 10 **/

data _acsvar(keep=schtmc _gpacs);
     set _acsvar end=eof;
     retain _jkrank 0;
     _jkrank+1;
     _gpacs=ceil(jkrank/10);
     if eof then call symput('_ngpacs',compress(put(_gpacs,8.),' ');
run;

proc sort data=statemp;
     by schtmc;
run;

/** To attach the grouping variable, _GPACS, to the STATEMP data    **/

data statemp;
     merge statemp _acsvar;
     by schtmc;
run;
```

TITLE AND FOOTNOTE ALIGNMENT

In the SAS/BASE environment, one procedure step can only declare one alignment -- left-justification, right-justification or center. That means all title and footnote texts are aligned either on the right, on the left, or in the center. In general, a mixed use of these three alignments is demanded in the pharmaceutical area and it would be appreciated if its functionality could be provided. Since the functionality is not yet available in the SAS system, TABRPT uses leading blanks to accomplish the requests of the right and center alignments. If a title or footnote is stated to be right-justified, j=r, a certain amount of leading blanks will be added prior to its text. This amount is calculated using the difference in length of the row and the text string. If specified to be in the center, j=c, the amount of leading blanks is half of that required for the right alignment. Below is an example of justifying a center alignment.

```
%let title=title had not been defined yet;

options nocenter ls=132 ps=45;

data _null_;
       _len=(132-length("&title"))/2;
       call symput('title',repeat(' ',_len)||"&title");
run;
```

PAGE BREAK

The most complicated task of developing TABRPT is the page break. The codes in TABRPT to deal with page breaking are too lengthy and complex to get into details. Therefore, this paper will only outline its strategy. First is to figure out the start position of a row on the page to display the contents and a maximum number of available rows on a page. For an illustration and the convenience, the start position and the maximum number of available rows are named MIN and MAX, respectively. MIN is the position of the second solid line on the report, which is also the number of rows used for title and header. MAX is the number of rows left after deducting those occupied by titles, headers, footnotes, the pagination line, and 3 solid lines from total rows of the page. Next is to create a row numbering variable, _obsln, and a page numbering variable, _pagno. The row numbering indicates an observation's row position on a page and it ranges from MIN+1 to MIN+MAX. The row numbering values are not a sequence of consecutive numbers. They vary depending on additional lines stated within compute blocks, created by break statements, or flow option in a define statement. A new page will begin when the value of the PAGE variable, if any, changes or when the row numbering refreshes with the start number, MIN. The PAGE variable is created by a break statement. The row numbering re-starts only if its value or its value adding rows used by the next group of a Break value, if defined, is greater than MIX+MAX.

When an ACROSS variable is specified, only observations in the first ACROSS group will be processed through those steps mentioned above to obtain the page numbering and row numbering variables. Next, the first ACROSS group will share its page and row numberings with the rest of groups and the shared page numbering will be rearranged in an appropriate order. This rearrangement will assure all ACROSS values will be displayed on consecutive pages. Below is an example of the creation of the page and row numbering variable and variables of additional lines being printed before and after each observation. These codes were generated by TABPRT using the data described in the section of usage of TABRPT.

```
/** To obtain observations in the first ACROSS group                                    **/

proc sort data=statemp(where=(_gpacs=1)) out=_jkrpt;
     by analyc trt dummy ptno prd schtmc;
run;

/** To posscess the page break and create the following variables                       **/
/** 1) Row numbering named _obsln, 2) Page numbering named _pagnum, 3)Additional lines being printed**/
/** before each obs. named _bw, 4) Additional lines being printed after each obs. named _aw.        **/
/** MIN is 16 and MAX is 34. The Page variable is analyc. The BREAK is trt.  The ACROSS is schtmc.  **/

data _jkrpt;
     set _jkrpt end=eof;
     by analyc trt dummy ptno prd schtmc;
     retain _obsln 16 _pagcon _paggln0 _pagln1 _pagnum 0;
     _bw=sum(_exbln1, _exbln2, _exbln3, 0);
     _aw=sum(_exaln1, _exaln2, _exaln3, 0);
     if _n_ eq 1 then do;
        _pagln0=sum(0, _exlin1, _exlin2, _exlin3);
        _pagln1=sum(0, _exlin1);
        call symput('_bw',compress(put(_bw,8.),' '));
     end;
     if first.analyc then do;
        _pagnum+1;
        _obsln=16+(_pagln0-sum(0, _exlin1, _exlin2, _exlin3));
        _pagcon=0;
     end;
     if first.trt then do;
        if (0 lt sum(_obsln+max(0, _brkln1))-(16+34) le _modbrk) or
           (0 le (16+34-obsln) le _modbrk) then do;
           if (_pagln1-sum(0, _exlin1) eq 0) or (0 le (16+34-_obsln) le _modbrk then _pagcon=0;
           else _pagcon=1;
           if _obsln ne 16+1 then do;
              _pagnum+1;
              _obsln=16+(_pagln1-sum(0, _exlin1));
              if _pagcon eq 1 then _obsln+1;
           end;
        end;
        else if _brkln1 > 34 or (sum(_obsln+max(0, _brkln1))-(16+34) gt _modbrk) then _pagcon=1;
     end;
     if sum(_obsln, _exlin1, _exlin2, _exlin3, _lines) > (16+34) then do;
        _pagnum+1;
        _obsln=16+_pagln0-sum(0, _exlin1, _exlin2, _exlin3);
        if _pagcon=1 then _obsln+1;
```

```
     end;
     _obsln=sum(_obsln, _exlin1, _exlin2, _exlin3, _lines);
run;

/** To share the page and row numbering and additional lines needed before & after each observation **/

data _jkrpt(drop=_jkbw);
     set _jkrpt(rename=(_bw=_jkbw));
     by analyc trt dummy ptno prd;
     retain _bw 0;
     if first.prd then _bw=jkbw;
     if last.prd;
run;

data statemp;
     merge _jkrpt statemp;
     by analyc trt dumy ptno prd;
run;

/** 1) Rearrange the page numbering in an appropriate order.                                    **/
/** 2) Create a macro variable containing the number of total pages of the report.              **/

DATA statemp;
     SET statemp;
     Length _pagno $ 7;
     _pagtmp=_pagnum+(floor(_pagnum)-1)*(2-1)+(_gpacs-1)+(_pagcon)/10;
     _pagno=put(_pagtmp,7.1);
     call symput('_lstpag',compress(put(floor(_pagtmp),8.),' '));
run;
```

REPORT GENERATION

After values of the page and row numbering variables and the additional lines needed are obtained, there are three steps needed to generate a final report. These steps are a PROC REPORT, a DATA step, and a DATA _NULL_ with the file print statement. Proc report is the main step to create a draft report that contains additional columns for the row and page numbering information and the additional lines needed before and after each observation on the report, but does not include any footnotes. It also outputs the draft report to an external text file. The data step extracts the report, the row and page numbering, and additional line needed information from the text file. It creates a SAS data set using the extracted information. This data set contains three variables. The first variable, _txt, contains the text of each row of the final report. The second and third variables contain the page and row numbering information for each row. The DATA _NULL_ step puts each value of _txt in the report according to its associated page and row numbering values. In addition, it processes the printing of any conditional footnotes, outputs conditional and unconditional footnotes, and displays the customized pagination onto the right corner of the last row of a page. Because of the complexity of coding, here only examples of PROC REPORT and DATA _NULL_ steps created by TABRPT are provided using the data described in the section of usage of TABRPT.

```
/*** proc report procedure ****/

%let _undl=%sysfunc(repeat(_,%eval(&_ls-1)));

options ls=%eval(132+20) nobyline;

proc sort data=statemp(rename=(_obsln=_rw));
     by _gpacs _pagno analyc trt dummy ptno prd schtmc;
run;

proc printto print="_junk.txt" new;
run;

proc report data=statemp split='^' ps=45 nocenter nowd headline missing;
     by _gpacs;
     column ("&_undl" trt dummy ptno prd schtmc,(charrslt condft1) _pagno _rw _bw _aw);
     define _pagno /group spacing=1 width=7;
     define _rw    /mean spacing=1 format=3. width=3;
     define _bw    /mean spacing=1 format=3. Width=3;
     define _aw    /mean spacing=1 format=3. Width=3;
     define analyc /group noprint;
     define trt    /group noprint;
     define dummy  /group odrer=internal noprint;
     define schtmc /across order=data format=gtform. "" center;
     define ptno   /group order=data format=subform. width=10 "Subject" center;
     define prd    /group width=7 "Period" CENTER;
     efine charrslt/display width=8 spacing=2 "" center nozero;
     define condft1/display format=cf. width=1 spacing=0 "" center nozero;
     compute before analyc;
        line @3 "ANALYTE:  " analyc analyfmt.;
```

```
     endcomp;
     compute before trt;
        line @3 'REGIMEN:  ' trt treat.;
        line "";
     endcomp;
     break after analyc/page;
     break after trt   /skip;
     break after dummy /skip;
title1 "  DUMMY STUDY: STUDY 99-999";
title2 "  18JAN2000 <SUGI25.SAS YENP>";
title3 "  SAMPLE EXAMPLE REPORT";
title5 "                                                                SUGI 25";
title7 "                                                    REPORT GENERATED BY TABRPT";

printto;
run;

/** The data _null_ step **/

proc sort data=_out;
     by _pagno _jkrow;
run;

options ls=132 byline;

data _null_;
     set _out end=eof;
     by _pagno;
     file print notitles;
     if first._pagno then put _page_;
     put #_jkrow @1 _txt $132.;
     if last._pagno then link foot;
     return;
foot:  /** processing the printing of conditional footnotes **/
     _jkpage+1;
     put @1 _jkx $132.;
     length _cond $ 1;
     _cond=symget('_cond'||compress(put(floor(_pagno),8.),' '));
     do _ii=1 to 1 by -1;
        _jkx=symget('_coln'||compress(put(_ii,8.),' '))||
             symget('_coc'||compress(put(_ii,8.),' '));
        if substr(_cond,_ii,1) eq '1' then put @(3-1) _jkx $129.;
     end;
     put #54 @(132-15) 'Page' _jkpage " of 8";
     return;
run;
```

CONCLUSION

The macro, TABRPT, has been an extremely useful report generator for TAP. It not only saves times of SAS coders when creating a customized report, but also motives them to understand the SAS/BASE functions and statements more. In addition, it has gradually advanced the coders to see the SAS language as another kind of computer language.

REFERENCE

1. SAS Institute Inc. (1990), *SAS Guide to Macro Processing, Version 6, Second Edition,* Cary, NC: SAS Institute Inc.
2. SAS Institute Inc. (1990), *SAS Guide to the REPORT Procedure, Usage and Reference, Version 6, First Edition,* Cary, NC: SAS Institute Inc.
3. SAS Institute Inc. (1990), *SAS Language, Reference, Version 6, First Edition,* Cary, NC: SAS Institute Inc.
4. SAS Institute Inc. (1990), *SAS Procedures Guide, Version 6, Third Edition,* Cary, NC: SAS Institute Inc.
5. SAS Institute Inc. (1997), *SAS Macro Language, Reference, Version 6, First Edition,* Cary, NC: SAS Institute Inc.

ACKNOWLEDGMENTS

A deep appreciation goes to those at TAP, who had or has being involved with TABRPT, for their contributions and feedback during this macro development.

CONTACT INFORMATION

Because of the paper page limitation, TABRPT's code is not provided here. However, if you would like to know more about it, please feel free to contact us and we will be happy to discuss it with you. Your comments and questions are valued and encouraged.

Peng-fang Yen
Atlantic Searching Group
10932 75th St. #204
Kenosha, WI 53142
Work:(847) 236-2127
Fax: (847) 236-2403
Email:peng-fang.yen
@tappharma.com
Web:www.asg-inc.com

Jeff Gudmundson
TAP Holdings Inc.
2355 Waukegan Rd.
Deerfield, IL 60015
Work:(847)267-4935
Fax: (847)267-5600
Email:jeff.gudmundson
@tappharma.com
Web:www.tappholdings.com

DUMMY DRUG: STUDY 99-999
18JAN2000 <SUGI25.SAS YENP>
SAMPLE EXAMPLE REPORT

SUGI 25

REPORT GENERATED BY TABRPT

Subject	Period	0 hr	0.5 hr	1 hr	1.5 hr	2 hrs	3 hrs	4 hrs	6 hrs	8 hrs	12 hrs
ANALYTE:	DUMMY DRUG										
REGIMEN:	TRTMENT-1										
201	2	0.00	0.00	25.10	116.00	366.00	1220.00	2000.00	1540.00	1600.00	1300.00
203	1	0.00	0.00	0.00	53.80	283.00	1010.00	2170.00	2090.00	1900.00	1510.00
204	2	0.00	0.00	85.80	547.00	1280.00	1670.00	1700.00	1670.00	1570.00	1120.00
206	1	0.00	0.00	0.00	13.90	55.10	204.00	278.00	1310.00	1910.00	1310.00
207	1	0.00	0.00	22.50	169.00	1100.00	1830.00	2040.00	2110.00	1830.00	1360.00
208	2	0.00	0.00	0.00	32.40	219.00	1190.00	1550.00	1840.00	1900.00	1490.00
209	1	0.00	0.00	12.00	63.30	490.00	1370.00	1630.00	1560.00	1600.00	1380.00
210	2	0.00	0.00	134.00	285.00	574.00	1430.00	1550.00	1990.00	1880.00	1440.00
212	2	0.00	10.40	25.70	139.00	399.00	729.00	1290.00	1050.00	1840.00	1050.00
213	1	0.00	0.00	204.00	637.00	757.00	1390.00	1930.00	1720.00	2130.00	938.00
214	2	0.00	16.40	21.80	41.60	106.00	1290.00	2520.00*	1960.00	2810.00*	1970.00
215	1	0.00	0.00	37.80	233.00	510.00	940.00	1300.00	1380.00	1540.00	1200.00
216	2	0.00	22.50	103.00	319.00	1320.00	1660.00	2260.00	1560.00	1360.00	674.00
217	1	0.00	0.00	12.10	238.00	1190.00	602.00	2300.00	1950.00	1930.00	1470.00
219	1	0.00	0.00	10.50	13.60	30.00	167.00	1200.00	1510.00	1220.00	920.00
220	2	0.00	10.50	53.50	324.00	825.00	1490.00	1630.00	1390.00	1380.00	1020.00
N		16	16	16	16	16	16	16	16	16	16
Mean		0.00	3.74	46.74	201.60	594.01	1137.00	1709.25	1664.38	1775.00	1259.50
Median		0.00	0.00	23.80	154.00	500.00	1255.00	1665.00	1615.00	1835.00	1305.00
S.D.		0.00	7.17	57.48	187.08	438.26	497.91	546.44	306.48	372.72	306.82
CV (%)			191.71	122.98	92.80	73.78	43.79	31.97	18.41	21.00	24.36
Minimum		0.00	0.00	0.00	13.60	30.00	167.00	278.00	1050.00	1220.00	674.00
Maximum		0.00	22.50	204.00	637.00	1320.00	1830.00	2520.00*	2110.00	2810.00*	1970.00

* indicating value > 2500.

TABLE 1