PharmaSUG 2020 - Paper QT-213

A SAS Macro for Dynamic Assignment of Page Numbers

Manohar Modem, Cytel Inc;
Bhavana Bommisetty, Vita Data Sciences;

ABSTRACT

In clinical domain, we usually create many safety and efficacy tables with various statistics. While creating these tables, we introduce SAS dataset with statistics into PROC REPORT to create listing or rtf output. Using PROC REPORT-BREAK-PAGE, we can make sure that each parameter statistics starts in a new page in the output. If we want to make sure that a group of statistics does not break abruptly between pages, we may need to use conditional statements to assign page numbers. Whenever there is an update in mock shell or data, the number of rows in the output may increase or decrease, which in turn requires an update in conditional statements to prevent abrupt, breaks in the output. This led to an effort to create a macro, which prevents abrupt page breaks and provides meaningful page numbers. This paper describes how the page numbers were dynamically assigned using SAS macro.

INTRODUCTION

To create a table, we generate descriptive or inferential statistics using various SAS procedures and present them as per mock shell. We introduce the final dataset, which contains all the statistics in the required format into PROC REPORT procedure to create rtf output. In most cases, we may have to create a custom variable using IF-THEN-ELSE conditional statements and introduce this in PROC REPORT-BREAK-PAGE to avoid abrupt breaks in the output.

Figure 1 and Figure 2 shows an abrupt break in a demographics table output where ‘Height’ statistics are shown in two different pages, as page 1 cannot accommodate any more rows. Figure 3 and Figure 4 shows that ‘Height' statistics are pushed to page 2. We can do this by creating variable using IF-THEN-ELSE conditional statements and using this variable in PROC REPORT.

	Placebo	Active
Age (year)		
n	3	13
mean (SD)	14.3 (0.58)	12.8 (1.52)
median	14.0	12.0
min, max	14, 15	11, 16
Sex		
Male	2 (67)	6 (46)
Female	1 (33)	7 (54)
Race		
Caucasian	1 (33)	3 (23)
Black	0	2 (15)
Hispanic	0	2 (15)
Asian	0	2 (15)
Other	2 (67)	4 (31)
Height (cm)		
n	3	13
mean (SD)	65.0 (3.50)	61.1 (5.57)

Figure 1. Demographics table - Page 1 of 2

	Placebo	Active
median	63.5	59.8
min, max	63, 69	51, 72
Weight (Kg)		
n	3	13
mean (SD)	109.2 (5.77)	97.3 (26.42)
median	112.5	90.0
min, max	103, 113	51, 150

Figure 2. Demographics table - Page 2 of 2

	Placebo	Active
Age (year)		
n	3	13
mean (SD)	14.3 (0.58)	12.8 (1.52)
median	14.0	12.0
min, max	14, 15	11, 16
Sex		
Male	2 (67)	6 (46)
Female	1 (33)	7 (54)
Race		
Caucasian	1 (33)	3 (23)
Black	0	2 (15)
Hispanic	0	2 (15)
Asian	0	2 (15)
Other	2 (67)	4 (31)

Figure 3. Demographics table - Page 1 of 2

	Placebo	Active
Height (cm)		
n	3	13
mean (SD)	65.0 (3.50)	61.1 (5.57)
median	63.5	59.8
min, max	63, 69	51, 72
Weight (Kg)		
n	3	13
mean (SD)	109.2 (5.77)	97.3 (26.42)
median	112.5	90.0
min, max	103, 113	51, 150

Figure 4. Demographics table - Page 2 of 2

In clinical domain, there are different kinds of table like demographic, lab shift, change from baseline etc., which have different shell structure. So, for each of these tables, you may need to create a custom variable using IF-THEN-ELSE conditional statements to avoid abrupt page breaks. This led to an effort to create %pageno macro that can be used across various tables.

%PAGENO MACRO DESCRIPTION

The functionality of this macro is to create a numeric variable, which prevents abrupt page breaks in the output.

This macro contains six keyword macro parameters.

- &indata = name of the input dataset. By default, this macro considers the latest created dataset before the macro call.
- &outdata = name of the output dataset. By default, this macro creates an output dataset named “pg__final”.
- &mxlnpg = maximum number of rows desired per page in the rtf or listing output.
- &grpvars = list of grouping variables separated by space. It can be one to many character or numeric variables. A variable can be given only once as this macro parameter value. See Figure 8 for more information.
- &statvar = variable with statistic labels or its corresponding numeric variable. This parameter can have only one variable.
- &debug = possible values are 0 and 1. Default value is 0. &debug=1 generates all the intermediate datasets.

&mxlnpg, &grpvars and &statvar are the three macro parameters that are mandatory for each call. We can use other parameters as and when required.

Usually in a table rtf output, each page consists of three parts - Title, Footnotes and Body. The number of lines occupied by title and footnotes varies from one table to another. As a result, the number of lines available in the body is different for various kind of tables.

Let us consider the examples of Demographics table and Summary of Vital signs table to understand the functionality of %pageno macro.

Parameter	Summary	Placebo (N=xxx)	Dose 30 mg (N=xxx)
Age (year)			
	n	xxx	xxx
	mean (SD)	xx (x)	xx (x)
	median	xxx	xxx
	range	(xxx, xx)	(xxx, xx)
Sex			
	Male	xxx (xx)	xxx (xx)
	Female		
Race			
	Caucasian	xxx (xx)	xxx (xx)
	Black	xxx (xx)	xxx (xx)
	Hispanic	xxx (xx)	xxx (xx)
	Asian	xxx (xx)	xxx (xx)
	Other	xxx (xx)	xxx (xx)
Height (cm)			
	n	xxx	xxx
	mean (SD)	xxx (xx.x)	xxx (xx.x)
	median	xxx	xxx
	range	(xxx, xx)	(xxx, xx)
Weight (kg)			
	n	xxx	xxx
	mean (SD)	xxx.x	xxx.x
	median	xxx	xxx
	range	(xxx, xx)	(xxx, xx)

Figure 5. Mock shell for Demographics

	SECTION	ORDER	TEXT	PLACEBO	ACTIVE	PAGE_F
1	1	0	Age (year)			1
2	1	1	n	3	13	1
3	1	2	mean (SD)	14 (1)	13 (2)	1
4	1	3	median	14	12	1
5	1	4	range	(14, 15)	(11, 16)	1
6	2	0	Sex			1
7	2	1	Male	2 (67)	6 (46)	1
8	2	2	Female	1 (33)	7 (54)	1
9	3	0	Race			2
10	3	1	Caucasian	1 (33)	3 (23)	2
11	3	2	Black	0	2 (15)	2
12	3	3	Hispanic	0	2 (15)	2
13	3	4	Asian	0	2 (15)	2
14	3	5	Other	2 (67)	4 (31)	2
15	4	0	Height (cm)			2
16	4	1	n	3	13	2
17	4	2	mean (SD)	65 (3.5)	61 (5.6)	2
18	4	3	median	64	60	2
19	4	4	range	(63, 69)	(51, 72)	2
20	5	0	Weight (Kg)			3
21	5	1	n	3	13	3
22	5	2	mean (SD)	109 (5.8)	97 (26.4)	3
23	5	3	median	113	90	3
24	5	4	range	(103, 113)	(51, 150)	3
25	6	0	Body Mass Index			3
26	6	1	n	3	13	3
27	6	2	mean (SD)	18.2 (1.8)	18.0 (2.3)	3
28	6	3	median	18	18	3
29	6	4	range	(17, 20)	(13, 21)	3

Figure 6. Demographics - final SAS dataset

SAS code:

```
%pageno (grpvars= section, statvar= order, mxlnpg= 12)
```

Figure 5 represents a sample for Demographics mock shell.

Figure 6 shows SAS dataset, which is generated as per Demographics mock shell in Figure 5. In this dataset, 'page_f' variable is created using above %pageno macro call. In this macro call, &grpvars= section. 'Section' is a numeric grouping variable, which presents sets of rows like 'Age', 'Sex', 'Race' etc in the required sorting order. 'Order' is a numeric variable, which represents the soring order in each set of rows in 'text' variable. &mxlnpg parameter value is set to 12, which means the maximum number of records that we would like to see in each page is 12. At 12th record, text = 'Hispanic'. This record is not the last record in 'Race' section. Therefore, macro pushes all 'Race' section records to next page where 'page_f' value is 2. Again, macro continues to check for 12th record starting from first row of 'Race' section. So, at the next 12th record, text = 'Weight(kg)' and it is not the last record of 'Weight(kg)' section. This section gets pushed to next page where 'page_f' variable value is 3. In this way, macro assigns numeric values to 'page_f' variable until the end of the dataset. We can use this 'page_f' variable in PROC REPORT-BREAK-PAGE as shown below to prevent abrupt page breaks.

SAS Code:

```
proc report;
    column PAGE_F section order text placebo active;
    define page_f/order order=internal noprint;
    define section/ order order=internal noprint;
    define order/ order order=internal noprint;
    define text/ display;
    define placebo/ display;
    define active/display;
    break after PAGE_F/page;
    break after section/skip;
  run;
```

Parameter: Diastolic Blood Pressure

	Placebo	Active
Baseline		
n	xx	xx
Mean	xx.x	xx.x
SD	xx.xx	xx.xx
Median	xx.x	xx.x
Min, Max	xx.x, xx.x	xx.x, xx.x
Week 1		
n	xx	xx
Mean	xx.x	xx.x
SD	xx.xx	xx.xx
Median	xx.x	xx.x
Min, Max	xx.x, xx.x	xx.x, xx.x
Continue to do: till last week		

Figure 7. Mock shell for Summary of Vital signs

Figure 7 represents a sample for Summary of Vital signs mock shell.

	PARAMCD	AVISITN	AVISIT	ORDER	TEXT	PLACEBO	ACTIVE	PAGE_F
1	DIABP	0	Baseline	0	Baseline			1
2	DIABP	0	Baseline	1	n	19	25	1
3	DIABP	0	Baseline	2	Mean(SD)	80 (8.54)	80 (8.09)	1
4	DIABP	0	Baseline	3	Median	81.0	80.0	1
5	DIABP	0	Baseline	4	Q1, Q3	71.0, 87.0	74.0, 84.0	1
6	DIABP	0	Baseline	5	Min, Max	67.0, 97.0	65.0, 96.0	1
7	DIABP	1	Week 1	0	Week 1			1
8	DIABP	1	Week 1	1	n	19	25	1
9	DIABP	1	Week 1	2	Mean(SD)	80 (7.02)	81 (8.72)	1
10	DIABP	1	Week 1	3	Median	79.0	82.0	1
11	DIABP	1	Week 1	4	Q1, Q3	73.0, 86.0	74.0, 88.0	1
12	DIABP	1	Week 1	5	Min, Max	68.0, 93.0	62.0, 97.0	1
13	DIABP	2	Week 2	0	Week 2			2
14	DIABP	2	Week 2	1	n	19	25	2
15	DIABP	2	Week 2	2	Mean(SD)	80 (8.17)	82 (9.94)	2
16	DIABP	2	Week 2	3	Median	79.0	82.0	2
17	DIABP	2	Week 2	4	Q1, Q3	76.0, 84.0	76.0, 90.0	2
18	DIABP	2	Week 2	5	Min, Max	60.0, 97.0	63.0, 98.0	2
19	SYSBP	0	Baseline	0	Baseline			3
20	SYSBP	0	Baseline	1	n	19	25	3
21	SYSBP	0	Baseline	2	Mean(SD)	126 (11.15)	127 (13.91)	3
22	SYSBP	0	Baseline	3	Median	127	125	3
23	SYSBP	0	Baseline	4	Q1, Q3	119, 134	119, 132	3
24	SYSBP	0	Baseline	5	Min, Max	107, 147	95.0, 159	3
25	SYSBP	1	Week 1	0	Week 1			3
26	SYSBP	1	Week 1	1	n	19	25	3
27	SYSBP	1	Week 1	2	Mean(SD)	127 (9.33)	128 (12.24)	3
28	SYSBP	1	Week 1	3	Median	127	125	3
29	SYSBP	1	Week 1	4	Q1, Q3	120, 132	118, 136	3
30	SYSBP	1	Week 1	5	Min, Max	112, 147	107, 152	3
31	SYSBP	2	Week 2	0	Week 2			4
32	SYSBP	2	Week 2	1	n	19	25	4
33	SYSBP	2	Week 2	2	Mean(SD)	125 (15.59)	127 (11.61)	4
34	SYSBP	2	Week 2	3	Median	123	124	4
35	SYSBP	2	Week 2	4	Q1, Q3	116, 131	118, 132	4
36	SYSBP	2	Week 2	5	Min, Max	95.0, 156	110, 154	4

Figure 8. Summary of Vital signs - final SAS dataset

SAS code:

```
%pageno (grpvars= paramcd avisitn, statvar= order, mxlnpg=15)
```

Figure 8 shows SAS dataset, which is generated as per Summary of Vital signs mock shell in Figure 7. In the above %pageno macro call, &mxlnpg is set to 15. At row 15, PARAMCD=DIABP, AVISITN = 2, and ORDER=2. As this is not the last record at PARAMCD = DIABP and AVISITN=2, rows in AVISITN =2 are pushed to next page where 'page_f' = 2. Then you notice that even though (PARAMCD = DIABP and AVISITN = 2) and (PARAMCD = SYSBP and AVISITN=0) constitute less than 15 records, these two sets of records have different 'page_f' value. This is because we want to have a page break at the beginning of every parameter in order to meet the shell requirements. In addition, a variable can be represented only once in &grpvars macro variable. For example, you cannot use both AVISIT and AVISITN in &grpvars, use either one of them.

CONCLUSION

By using this macro, we do not need to worry about abrupt breaks in the output either in the first-run or for every data update. This could be quite useful when the table output is too long or when table mock shell is complex with different number of rows in each set.

CONTACT INFORMATION

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

Manohar Modem
Sr. Statistical Programmer
Cytel Inc
1050 Winter St, Suite 2700
Waltham, MA/ 02451
Email: manohar.modem@gmail.com

Bhavana Bommisetty
Statistical Programmer
Vita Data Sciences
281 Winter St, Suite 100
Waltham, MA/ 02451
Email: bhavana.bommisetty@gmail.com

APPENDIX

```
%macro pageno (indata=_last_, outdata=pg__final, mxlnpg= , grpvars= ,
statvar= , debug=0);

%local grpvars_1 grpvars_2 grpvars_cs;
%let grpvars_1 = %scan(&grpvars, -1, " ");
%put grpvars_1 = **&grpvars_1**;
%let grpvars_2 = %scan(&grpvars, -2, " ");
%put grpvars_2 = **&grpvars_2**;
%if %bquote(&grpvars_2) = %str( ) %then %do;
 %let grpvars_2= __cat;
 %put grpvars_2 = **&grpvars_2**;
%end;
%let grpvars_cs =
%sysfunc(translate(%sysfunc(compbl(%sysfunc(strip(&grpvars)))), ",", " "));
%put grpvars_cs = **&grpvars_cs**;

data __dset0;
 set &indata;
 __cat=1;
run;

proc sort;
 by __cat &grpvars &statvar;
run;

proc sql noprint;
 select count(*) into :varchk
 from sashelp.vcolumn
 where upcase(libname) = 'WORK' and upcase(memname) = "__DSET0" and
upcase(name) = "PAGE_F";
quit;

data __dset1;
 set __dset0;
 by __cat &grpvars &statvar;
 if first.&grpvars_1 then pgvar_rec = 1;
 else pgvar_rec+1;
 totaln= _n_;
 if first.&grpvars_2 then __paramn0=1;
 __paramn+__paramn0;
 if pgvar_rec > &mxlnpg then do;
   put "WARNING: One or more categories has more records than %nrstr(&mxlnpg)
at: " &grpvars_1=
     &statvar=       pgvar_rec= ;
    put "WARNING: %nrstr(&mxlnpg) value should be >= pgvar_rec ";
  end;
 %if &varchk ^=0 %then %do;
  drop page_f;
 %end;
run;

data __dset2;
 set __dset1;
 by __cat &grpvars &statvar;
```

```
 retain remain_rec pg0 /*pg03*/;
 if first.&grpvars_2 then do;
  remain_rec=.;
  pg0=.;
 end;
 else do;
  if last.&grpvars_1 then do;
    if remain_rec=. then do;
      if pgvar_rec <= &mxlnpg then remain_rec = &mxlnpg - pgvar_rec;
      pg0 = 1;
    end;
     else if remain_rec ^=. then do;
     if remain_rec >= pgvar_rec then do;
      remain_rec= remain_rec - pgvar_rec ;
      pg0 +0;
     end;
     else if remain_rec < pgvar_rec then do;
      remain_rec= &mxlnpg - pgvar_rec ;
      pg0 +1;
     end;
   end;
  end;
 end;
run;

proc sql;
 create table __dset3 as
 select  *, max(pg0) as pg01
 from __dset2
 group by &grpvars_cs
 order by &grpvars_cs, &statvar
 ;
quit;

data __dset4;
 set __dset3;
 by __cat &grpvars ;
 retain page_f;
  dif1 = dif(pg01);
  if dif1>. then dif1=abs(dif1);
  if first.&grpvars_2 and dif1 = 0 then dif1=1;
  if __paramn=1 then page_f= pg01;
  else if dif1>0 then page_f+1;

run;

proc sort ;
 by &grpvars &statvar;
run;
data &outdata;
  set __dset4;
  drop totaln remain_rec pg0 pg01 pgvar_rec __: dif1;
run;

%if &debug ^= 1 %then %do ;
 proc datasets nolist;
  delete __: ;
```

```
 run;
%end;
%mend;

Sample call:
  %pageno ( grpvars= section, statvar= order, mxlnpg=12)
```