Paper 170-2008

Learning PROC REPORT by Comparison

Keith Cranford, Office of the Attorney General of Texas, Austin, Texas

ABSTRACT

You can use what you already know to learn PROC REPORT. This paper compares PROC REPORT to PROC PRINT, PROC TABULATE and PROC MEANS. Through the use of examples that produce similar reports, the syntax and features of PROC REPORT are illustrated.

PROC PRINT COMPARISON

You all probably use PROC PRINT to obtain simple data listings. In general, PROC PRINT is simple to use and provides a quick approach with minimal statements. However, beyond simple listings, PROC PRINT has limited options, whereas PROC REPORT can also produce simple listings, without too many more statements or options than PROC PRINT, and provides much more flexibility.

SIMPLE LISTING

The following statements can be used to produce a simple listing of SASHELP.CLASS.

```
title 'Simple Listing' ;
proc print data=sashelp.class ;
  var name sex age height weight ;
  title2 'PROC PRINT' ;
run ;
```

This code produces Output 1.A.

Output 1.A

Simple Listing
PROC PRINT

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83.0
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84.0
10	John	M	12	59.0	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90.0
13	Louise	F	12	56.3	77.0
14	Mary	F	15	66.5	112.0
15	Philip	M	16	72.0	150.0
16	Robert	M	12	64.8	128.0
17	Ronald	M	15	67.0	133.0
18	Thomas	M	11	57.5	85.0
19	William	M	15	66.5	112.0

The output in Output 1.A could also have been obtained by removing the VAR statement. In this case, PROC PRINT displays all variables in the data set. Note that an observation number is displayed and the data is formatted in a consistent manner in each column.

This report could be duplicated with PROC REPORT.

```
proc report data=sashelp.class nowd headskip ;
  column name sex age height weight ;
  define sex / width=3 ;
  title2 'PROC REPORT' ;
run ;
```

The COLUMN statement works much the same as the VAR statement in PROC PRINT. This indicates the order to display the columns in the report. The DEFINE statement provides column specific options. In this example, width=3 for the column SEX specifies the number of characters to be used, overriding the default width equal to the variable length. Otherwise, the column heading would be displayed vertically. Finally, HEADSKIP on the PROC REPORT statement skips a line after the heading, and NOWD indicates to not use the PROC REPORT windowing environment. This code produces the following output.

Output 1.B

Simple Listing
PROC REPORT

Name	Sex	Age	Height	Weight
Alfred	M	14	69	112.5
Alice	F	13	56.5	84
Barbara	F	13	65.3	98
Carol	F	14	62.8	102.5
Henry	M	14	63.5	102.5
James	M	12	57.3	83
Jane	F	12	59.8	84.5
Janet	F	15	62.5	112.5
Jeffrey	M	13	62.5	84
John	M	12	59	99.5
Joyce	F	11	51.3	50.5
Judy	F	14	64.3	90
Louise	F	12	56.3	77
Mary	F	15	66.5	112
Philip	M	16	72	150
Robert	M	12	64.8	128
Ronald	M	15	67	133
Thomas	M	11	57.5	85
William	M	15	66.5	112

By default, PROC REPORT does not display an observation number column, which could be added with a computed variable. Also, note that the Height and Weight columns do not display consistently. This could be remedied by adding a DEFINE statement for each of these columns and specifying a format. The value of the SEX column could be centered with a CENTER option (RIGHT or LEFT justification options are also available). The following code illustrates these changes.

```
proc report data=sashelp.class nowd headskip ;
  column Obs name sex age height weight ;
  define sex / width=3 center ;
  define age / width=5 ;
  define height / format=6.1 ;
  define weight / format=6.1 ;
  define obs / computed ;
  compute obs ;
    count+1 ;
    obs=count ;
  endcomp ;
  title2 'PROC REPORT, again' ;
run ;
```

This produces a report nearly identical to Output 1.A. The computed column OBS is the first column displayed. The

COMPUTE block for OBS (beginning with the COMPUTE OBS and ending with the ENDCOMP statement) specifies how the column is populated. First, a counter, COUNT, is incremented with each pass through the data. The value of COUNT is retained similar to a DATA step. The current value of the counter is then assigned to OBS.

Output 1.C

Simple Listing
PROC REPORT, again

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83.0
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84.0
10	John	M	12	59.0	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90.0
13	Louise	F	12	56.3	77.0
14	Mary	F	15	66.5	112.0
15	Philip	M	16	72.0	150.0
16	Robert	M	12	64.8	128.0
17	Ronald	M	15	67.0	133.0
18	Thomas	M	11	57.5	85.0
19	William	M	15	66.5	112.0

SORTED LISTING

A sorted listing can be obtained with PROC PRINT by first sorting the data set and, then using a BY statement.

```
proc sort data=sashelp.class out=class ;
  by sex ;
run ;

title 'Sorted Listing' ;
proc print data=class ;
  var name age ;
  by sex ;
  id sex ;
  title2 'PROC PRINT' ;
run ;
```

The ID statement prints the value of SEX only on the first occurrence and, also, specifies that SEX is to be displayed in the first column.

Output 2.A

Sorted Listing
PROC PRINT

Sex	Name	Age
F	Alice	13
	Barbara	13
	Carol	14
	Jane	12
	Janet	15

```
                          Joyce       11
                          Judy        14
                          Louise      12
                          Mary        15

                    M     Alfred      14
                          Henry       14
                          James       12
                          Jeffrey     13
                          John        12
                          Philip      16
                          Robert      12
                          Ronald      15
                          Thomas      11
                          William     15
```

PROC REPORT can produce a similar sorted listing without first using a PROC SORT.

```
proc report data=sashelp.class nowd headskip ;
  column sex name age ;
  define sex / order width=3 ;
  break after sex / skip ;
  title2 'PROC REPORT' ;
run ;
```

The ORDER option on the DEFINE statement specifies that this variable is to be used to order the report. Additionally, the BREAK AFTER statement with the SKIP option has been added to display a blank line after each group of rows with the same value for the specified variable. This is done to mimic the behavior of PROC PRINT with the ID statement above.

Output 2.B

```
                          Sorted Listing
                           PROC REPORT

                    Sex   Name            Age

                    F     Alice            13
                          Barbara          13
                          Carol            14
                          Jane             12
                          Janet            15
                          Joyce            11
                          Judy             14
                          Louise           12
                          Mary             15

                    M     Alfred           14
                          Henry            14
                          James            12
                          Jeffrey          13
                          John             12
                          Philip           16
                          Robert           12
                          Ronald           15
                          Thomas           11
                          William          15
```

Output 2.B matches Output 2.A exactly, with PROC REPORT requiring fewer statements.

LISTING WITH SUBTOTALS

PROC PRINT allows you to add subtotals at the end of a BY group. This is done with the SUMBY statement.

```
title 'Subtotals' ;
proc print data=class ;
  var name age ;
  by sex ;
  id sex ;
  sumby sex ;
  title2 'PROC PRINT' ;
run ;
```

As you can surmise from the statement the SUMBY variable produces sums of all the numeric variables in the listing, as seen in Output 3.A. You have no control over which numeric variables are summed.

Output 3.A

```
                              Subtotals
                              PROC PRINT

                        Sex    Name       Age

                         F     Alice       13
                               Barbara     13
                               Carol       14
                               Jane        12
                               Janet       15
                               Joyce       11
                               Judy        14
                               Louise      12
                               Mary        15
                        ---                ---
                         F                 119

                         M     Alfred      14
                               Henry       14
                               James       12
                               Jeffrey     13
                               John        12
                               Philip      16
                               Robert      12
                               Ronald      15
                               Thomas      11
                               William     15
                        ---                ---
                         M                 134
                                           ===
                                           253
```

In this example, the sum of the ages for the females is 119 and the sum of the males is 134. The grand total is 253.

This can be duplicated in PROC REPORT using the BREAK AFTER and RBREAK AFTER statements.

```
proc report data=sashelp.class nowd ;
  column sex name age ;
  define sex / order width=3 ;
  define age / analysis sum format=4.;
  break before sex / skip ;
  break after sex / summarize ol ;
  rbreak after / summarize dol ;
  title2 'PROC REPORT' ;
run ;
```

The BREAK AFTER statement specifies to summarize all analysis variables, putting a line over the summary (OL option) after each value of SEX. Also, the BREAK BEFORE statement with the SKIP option skips a line before the detail lines for each value of SEX. The RBREAK AFTER statement produces a report break at the end giving an overall summary, putting a double overline (DOL option) over the summary. The ANALYSIS and SUM options on the DEFINE statement for age indicate that age is an analysis column and to sum this column in any summaries. You control which columns to summarize by specifying either ANALYSIS or DISPLAY options on the DEFINE statement; DISPLAY will only display the column values and not summarize. The ANALYSIS option is optional when specifying a statistic such as SUM. Also, SUM is the default statistic for an analysis column.

Output 3.B

```
                    Subtotals
                  PROC REPORT

Sex  Name       Age

F    Alice       13
     Barbara     13
     Carol       14
     Jane        12
     Janet       15
     Joyce       11
     Judy        14
     Louise      12
     Mary        15
___              ____
F                119

M    Alfred      14
     Henry       14
     James       12
     Jeffrey     13
     John        12
     Philip      16
     Robert      12
     Ronald      15
     Thomas      11
     William     15
___              ____
M                134
                 ====
                 253
```

This example also points out an obvious shortcoming with the PROC PRINT SUMBY statement. Subtotal is the only statistic available in PROC PRINT, whereas PROC REPORT provides a variety of statistics such as MEAN, STD and RANGE. In this example, the sum of the ages is not very meaningful. The following code produces the same report as above, except with means, or averages.

```
proc report data=sashelp.class nowd ;
  column sex sex2 name age ;
  define sex / order width=3 noprint ;
  define sex2 / computed 'Sex' ;
  define age / mean format=best4. ;
  break before sex / skip ;
  break after sex / summarize ol ;
  rbreak after / summarize dol ;
  compute sex2 / character length=15 ;
     sex2=sex ;
  endcomp ;
  compute after sex ;
    sex2='Average for ' || sex ;
```

```
  endcomp ;
  compute after ;
    sex2='Overall Average' ;
  endcomp ;
  title2 'PROC REPORT and more' ;
run ;
```

The MEAN option on the DEFINE statement for AGE specifies to display the mean in any summarizations. This is the only change that is necessary to change the report in Output 3.B. However, this example also introduces some other capabilities of PROC REPORT that illustrate its flexibility.

First, as illustrated earlier, you can introduce computed (new) columns in PROC REPORT. The SEX2 column is a computed column. Its placement in the COLUMN statement specifies its position in the report. The DEFINE statement for SEX2 indicates it is a computed variable and the column heading is 'Sex.' The COMPUTE block for SEX2 specifies how the column is to be constructed. We are merely assigning the value of SEX to SEX2. The reason for doing this is found in the COMPUTE AFTER blocks. Here the value of SEX2 is being changed to display 'Average for F', 'Average for M' and 'Overall Average.' The SEX column could not have been used for this, because this variable has a length of 1, so there is not enough room for the additional text. The CHARACTER and LENGTH options on the COMPUTE statement indicate the type and length of this column. Finally, to suppress the SEX column from being printed the NOPRINT option is used on its DEFINE statement. This column is still needed to order the rows in the table, though.

Output 3.C

Subtotals
PROC REPORT and more

Sex	Name	Age
F	Alice	13
	Barbara	13
	Carol	14
	Jane	12
	Janet	15
	Joyce	11
	Judy	14
	Louise	12
	Mary	15
Average for F		13.2
M	Alfred	14
	Henry	14
	James	12
	Jeffrey	13
	John	12
	Philip	16
	Robert	12
	Ronald	15
	Thomas	11
	William	15
Average for M		13.4
Overall Average		13.3

This example can be refined slightly. Instead of creating a new column, freestyle text can also be printed at the summary breaks.

```
proc format ;
  value $gender 'F'='females'
                'M'='males' ;
run ;

proc report data=sashelp.class nowd headline ;
  column sex name age ;
  define sex / order width=3 ;
  define age / mean  ;
  break after sex / skip ;
  compute after sex ;
    text='Average for ' || strip(put(sex, $gender.)) ||
              ' is ' || put(age.mean, 4.1) ;
       line ' ';
    line text $40. ;
  endcomp ;
  compute after ;
    line 'Overall Average is ' age.mean 4.1 ;
  endcomp ;
  title2 'PROC REPORT and more, again' ;
run ;
```

The LINE statement works similar to the PUT statement in the DATA step. In this example, text such as 'Average for females is xx.x' will be displayed. Note that an analysis variable must be specified by their column name and statistic, for example, age.mean for mean of age. The temporary variable TEXT is used to account for varying lengths of the formatted values and suppress extra blanks.

Output 3.D

```
                    Subtotals
              PROC REPORT and more, again

  Sex  Name                 Age
  ──────────────────────────────
  F    Alice                 13
       Barbara               13
       Carol                 14
       Jane                  12
       Janet                 15
       Joyce                 11
       Judy                  14
       Louise                12
       Mary                  15

             Average for females is 13.2

  M    Alfred                14
       Henry                 14
       James                 12
       Jeffrey               13
       John                  12
       Philip                16
       Robert                12
       Ronald                15
       Thomas                11
       William               15

             Average for males is 13.4

              Overall Average is 13.3
```

These are a few examples of how tasks typically done with PROC PRINT can also be done with PROC REPORT. There is the added advantage that PROC REPORT is much more flexible with far more available options. There are many more options in PROC REPORT which are not covered in this paper. Please refer to the documentation for PROC REPORT for the complete list. Suffice it to say, that PROC REPORT can do anything that PROC PRINT can do, only better!

PROC TABULATE COMPARISON

PROC REPORT can do many of the types of reports usually done with PROC TABULATE. Unlike PROC PRINT, there are some reports that PROC TABULATE can do that are not possible, or at least not practical, with PROC REPORT. Still, it is nice to know that PROC REPORT provides an alternative in many situations.

SUMMARY STATISTICS

PROC TABULATE has traditionally been used to display summary statistics with a controlled format. It has provided an alternative to PROC MEANS that has very little formatting capabilities. The report in PROC TABULATE is controlled by the TABLE statement. A summary of AGE by SEX can be produced by the following code.

```
title 'Summary Statistics' ;
proc tabulate data=sashelp.class ;
  class sex ;
  var age ;
  table sex, age*(n*f=4. mean*f=6.2) ;
  title2 'PROC TABULATE' ;
run ;
```

This produces a report with the number of observations and the mean age in each sex category.

Output 4.A

Summary Statistics
PROC TABULATE

	Age	
	N	Mean
Sex		
F	9	13.22
M	10	13.40

PROC REPORT can produce a similar report using SEX as a GROUP variable.

```
proc report data=sashelp.class nowd split='~' box ;
  column sex age age=avg_age ;
  define sex / group width=3 ;
  define age / n 'N' format=4.0 width=4;
  define avg_age / mean 'Average~Age' format=6.2 width=7;
  title2 'PROC REPORT' ;
run ;
```

The GROUP option on the DEFINE statement for SEX specifies that all the observations with the same value of SEX are to be used to calculate statistics. In this example, N (number of non-missing observations) and MEAN (Average) are computed. However, a variable can only be used once in the COLUMN or DEFINE statements, so an alias is set for the second use of AGE by specifying AGE=AVG_AGE on the COLUMN statement and then using AVG_AGE in a DEFINE statement. You can specify a column heading on the DEFINE statement by enclosing the heading in quotes. In this example, a split character, a tilde, is specified on the PROC REPORT statement with the SPLIT='~' option. This allows you to control how column headings are split. Finally, the BOX option on the PROC REPORT statement will put the report in a format similar to PROC TABULATE with lines around the report.

Output 4.B

Summary Statistics
PROC REPORT

Sex	N	Average Age
F	9	13.22
M	10	13.40

The resulting report is not identical to that in Output 4.A, but it is close and the information provided is the same. Removing the BOX option would result in the report in Output 4.C.

```
proc report data=sashelp.class nowd split='~' headline ;
  column sex age age=avg_age ;
  define sex / group width=3 ;
  define age / n 'N' format=4.0 width=4;
  define avg_age / mean 'Average~Age' format=6.2 width=7;
  title2 'PROC REPORT' ;
run ;
```

The HEADLINE option on the PROC REPORT statement adds a line under the headings.

Output 4.C

Summary Statistics
PROC REPORT

Sex	N	Average Age
F	9	13.22
M	10	13.40

Cross-Tabulation

PROC TABULATE is also used for producing cross-tabular frequency reports similar to those in PROC FREQ, but again it provides more control on the format of the report. The following code produces a simple cross-tabular frequency table of AGE and SEX.

```
options missing='0' ;
title 'Cross-tabulation' ;
proc tabulate data=sashelp.class ;
  class sex age ;
  table age, sex*n*f=4. ;
  title2 'PROC TABULATE' ;
run ;
```

The MISSING option on the OPTIONS statement is used to display a zero instead of the usual period, when there is a missing numeric value.

Output 5.A

Cross-tabulation
PROC TABULATE

	Sex	
	F	M
	N	N
Age		
11	1	1
12	2	3
13	2	1
14	2	2
15	2	2
16	0	1

PROC REPORT can produce a similar report, using SEX as an ACROSS variable.

```
proc report data=sashelp.class nowd headline ;
  column age sex ;
  define age / group ;
  define sex / across width=3 right ;
  title2 'PROC REPORT' ;
run ;
```

The ACROSS option on the DEFINE statement for SEX specifies that its values will be displayed across the top of the report as columns. Since there are no other analysis variables, PROC REPORT produces a frequency table.

Output 5.B

Cross-tabulation
PROC REPORT

	Sex	
Age	F	M
11	1	1
12	2	3
13	2	1
14	2	2
15	2	2
16	0	1

As in the previous example, the BOX option could make this report look more like a PROC TABULATE report. Otherwise, the report provides similar information to Output 5.A.

CROSS-TABULATION WITH SUMMARY

Many times you have the need to combine a cross-tabulation with a statistical summary. PROC TABULATE has traditionally been the place to turn for this type of report.

```
options missing='?';
title 'Cross-tabulation with Summary' ;
proc tabulate data=sashelp.class ;
  class sex age ;
  var height ;
  table age, sex*height*mean*f=6.1 ;
  title2 'PROC TABULATE' ;
run ;
```

This procedure gives the average height for the combinations of AGE and SEX.

Output 6.A

Cross-tabulation with Summary
PROC TABULATE

	Sex	
	F	M
	Height	Height
	Mean	Mean
Age		
11	51.3	57.5
12	58.1	60.4
13	60.9	62.5
14	63.6	66.3
15	64.5	66.8
16	?	72.0

PROC REPORT can produce a similar report by adding an analysis variable.

```
proc report data=sashelp.class nowd split='~' headline ;
  column age sex, height ;
  define age / group ;
  define sex / across '__Sex__' width=3 right ;
  define height / analysis mean 'Average~Height'
                  format=6.1 width=8 ;
  title2 'PROC REPORT' ;
run ;
```

Note that the analysis variable HEIGHT follows a comma in the COLUMN statement. The comma specifies that the columns following should be stacked under the values of the ACROSS variable.

Output 6.B

Cross-tabulation with Summary
PROC REPORT

	Sex	
	F	M
Age	Average Height	Average Height
11	51.3	57.5
12	58.1	60.4
13	60.9	62.5
14	63.6	66.3
15	64.5	66.8
16	?	72.0

Surrounding the column heading with two underscores has the effect of extending these underscores for the entire width of the column.

PERCENTAGES

PROC TABULATE is also useful when you need percentages calculated. The PCTN and PCTSUM statistics are used to compute percentages based on counts or sums.

```
title 'Percentages' ;
proc tabulate data=sashelp.class ;
  class sex ;
  table sex, n pctn<sex> ;
  title2 'PROC TABULATE' ;
run ;
```

This produces a table similar to one from PROC FREQ.

Output 7.A

Percentages
PROC TABULATE

	N	PctN
Sex		
F	9.00	47.37
M	10.00	52.63

Percentages can also be computed with PROC REPORT in a very similar manner to PROC TABULATE.

```
proc report data=sashelp.class nowd box ;
  column sex N pctn ;
  define sex / group width=3;
  define n / width=5 ;
  define pctn / '%' format=percent7.1 ;
  title2 'PROC REPORT' ;
run ;
```

The PCTN computes percentages just as in PROC TABULATE. The value, however, does not automatically display as a percent, so the format option specifies to display it as a percent.

Output 7.B

Percentages
PROC REPORT

Sex	N	%
F	9	47.4%
M	10	52.6%

From these examples you can see that PROC REPORT can produce summary tables, cross-tabulations as well as work with percent of totals, much as PROC TABULATE does. You also have the same flexibility of formatting values and column headings. Although PROC REPORT might not be able to do everything that PROC TABULATE, it comes pretty close.

PROC MEANS COMPARISON

PROC REPORT can also produce tables similar to PROC MEANS, with greater formatting capabilities. This is done with the use of automatic statistics columns.

SUMMARY STATISTICS

PROC MEANS is usually the procedure of choice when you need a statistical summary.

```
title 'Summary Statistics 2' ;
proc means data=sashelp.class mean std min max maxdec=1;
  class sex ;
  var height ;
  title2 'PROC MEANS' ;
run ;
```

The MEAN STD MIN MAX options specify which statistics to display, and the MAXDEC option specifies the maximum decimal place to display for all statistics. This is about the extent of the formatting capabilities in PROC MEANS.

Output 8.A

Summary Statistics 2
PROC MEANS

The MEANS Procedure

Analysis Variable : Height

Sex	N Obs	Mean	Std Dev	Minimum	Maximum
F	9	60.6	5.0	51.3	66.5
M	10	63.9	4.9	57.3	72.0

PROC REPORT provides an alternative, but with much more formatting options.

```
proc report data=sashelp.class nowd split='~' headline ;
  column sex height, (n mean std min max) ;
  define sex / group width=3 ;
```

```
  define height / '__Height__' ;
  define n / 'N' width=3 ;
  define mean / 'Mean' format=5.1 ;
  define std / 'Std~Dev' format=5.2 ;
  define min / 'Min' format=5.1 ;
  define max / 'Max' format=5.1 ;
  title2 'PROC REPORT' ;
run ;
```

Similar to PROC MEANS, PROC REPORT can use statistics for the report columns. Also, formats can be applied to individual columns, such as is done with STD. This provides more flexibility in formatting the report.

Output 8.B

Summary Statistics 2
PROC REPORT

	Height				
Sex	N	Mean	Std Dev	Min	Max
F	9	60.6	5.02	51.3	66.5
M	10	63.9	4.94	57.3	72.0

CONCLUSION

PROC REPORT can become your all-in-one solution for reporting needs. If you already know how to use PROC PRINT, PROC TABULATE and PROC MEANS, you can leverage that knowledge to produce similar reports with PROC REPORT, but with greater control on how the report appears.

CONTACT INFORMATION

Keith Cranford
Office of the Attorney General of Texas
Child Support Division
5500 E. Oltorf
Austin, TX 78741
keith.cranford@cs.oag.state.tx.us