

The Workhorses of Customized Report Writing – PROC TABULATE and PROC REPORT

Deborah Babcock Buck, D. B. & P. Associates, Houston, TX

ABSTRACT

SAS® has a number of procedures that produce summary reports. These can include reports developed for assistance in data cleansing, for descriptive statistics, or for statistical analyses (hypothesis testing, modeling, prediction, etc.) Two of the Base SAS procedures that are very versatile in generating summary reports are PROC TABULATE and PROC REPORT.

This paper attempts to introduce you to some of the capabilities of both of these procedures and to provide some guidance on which is the more appropriate for a given need. An example of the output from each of these procedures and the code that produced it are included. The code presented in this paper is valid for SAS Versions 6-9.

INTRODUCTION

Base SAS has a number of procedures to help you report your data in the form you would like to see it presented. PROC TABULATE and PROC REPORT are what I consider to be the ‘workhorses’ of report writing. Which of these two is more appropriate is based on your needs for a given situation. Which of the following do you need?

- Do you need additional columns of variable-type information?
- Do you need a customized look or ‘non-standard’ features?
- Do you need cross-tabulations or hierarchical groupings?
- Do you need statistics?

We will consider these items to see how TABULATE and REPORT can produce the desired summary. Examples of code and the resulting output will be presented with a detailed explanation of the code for each example.

DO YOU NEED ADDITIONAL COLUMNS OF VARIABLE-TYPE INFORMATION?

Frequently, PROC FREQ and/or PROC MEANS/SUMMARY are worth considering for a summary report, especially if you’re using the report primarily for obtaining quick information or data cleaning. However, if you need the capability to customize the report or add additional rows/columns of information, TABULATE or REPORT is probably more useful.

Do you need additional columns of information in the report that do not already exist as variables in the data set? An example of this might be the sum or average of two variables in the data set. If that information is only necessary for the report, why go through an additional DATA step to create a new variable? REPORT will allow you to generate new columns of information within the PROC step.

If you need the same information represented in more than one way in a report (for example, a medical code represented as a coded value from the data set as well as a formatted value), REPORT will allow you to show both columns in the same report. Some other procedures will display both columns, but require that you create a new variable for the formatted information. Using PROC FORMAT to create the format to associate with the coded value in conjunction with REPORT will allow you to use both the coded and formatted values in the same report without creating a new variable.

DO YOU NEED A CUSTOMIZED LOOK OR 'NON-STANDARD' FEATURES?

This could include, but is not limited to the following:

- providing customized information for subgroups or at the beginning or end of a report,
- underlining column headers,
- allowing variable values to extend across more than one line,
- outlining or not outlining the entire table and/or cells within the table.

TABULATE and REPORT have differing capabilities for doing the above items. We will look at ways of generating some of the above items later in this paper.

DO YOU NEED CROSS-TABULATIONS OR HIERARCHICAL GROUPINGS?

This type of table is a summary report where you display some sort of statistic(s) within a combination of variable values in the row and/or column dimensions. Although some Base SAS procedures, like PROC FREQ, can produce some reports of this type, TABULATE and/or REPORT are frequently used because of their greater versatility, flexibility and ease of use.

TABULATE is a natural choice for this type of report. If you don't need customized information included in your report, it should be your first consideration. On the other hand, if you need customized information within the report, you should also consider REPORT.

DO YOU NEED STATISTICS?

Both TABULATE and REPORT include a number of statistics. Both procedures include the following statistics (although the keywords may be slightly different).

N	MIN	USS	T	PCTN
NMISS	MAX	CSS	PRT	PCTNSUM
MEAN	RANGE	STDERR	VAR	
STD	SUM	CV	SUMWGT	

Additionally, TABULATE can produce the median and a number of percentiles.

In the following sections we will examine examples of TABULATE and REPORT and discuss the statements within each procedure to accomplish our goal reports. Both the output and the code used to generate the output will be presented. Each procedure assumes we are using an existing temporary SAS data set called PRESSURE, which includes the variables PTNO, RACE, GENDER, BP1 (Baseline Diastolic Blood Pressure), BP3 (Final Diastolic Blood Pressure), and CBP3 (Change in Final Diastolic Blood Pressure).

PROC TABULATE

The following output demonstrates some of the capabilities of PROC TABULATE.

TABLE 1
OUTPUT FROM PROC TABULATE

	Gender																	
	Female									Male								
	Baseline Diastolic BP			Final Diastolic BP			Change in Diastolic BP			Baseline Diastolic BP			Final Diastolic BP			Change in Diastolic BP		
	No.	Mean	Std Dev.	No.	Mean	Std Dev.	No.	Mean	Std Dev.	No.	Mean	Std Dev.	No.	Mean	Std Dev.	No.	Mean	Std Dev.
Race																		
White	44	94.70	5.52	44	92.73	8.00	44	-1.98	6.32	23	97.13	6.57	23	93.91	9.23	23	-3.22	6.79
Black	7	100.00	1.91	7	99.71	4.27	7	-0.29	5.35	4	98.50	5.45	4	97.50	4.80	4	-1.00	5.60
Other	2	101.00	4.24	2	96.50	9.19	2	-4.50	4.95	1	97.00	0	1	90.00	0	1	-7.00	0
All	53	95.64	5.51	53	93.79	7.91	53	-1.85	6.12	28	97.32	6.22	28	94.29	8.62	28	-3.04	6.50

The following code produced the above Table 1.

```

OPTIONS MISSING='0' NONUMBER NODATE LINESIZE=142 PAGESIZE=42;

/**** PROC TABULATE (TABLE 1)****/

PROC TABULATE DATA=PRESSURE NOSEPS;
CLASS GENDER RACE;
VAR BP1 BP3 CBP3;
TABLE (RACE ALL), GENDER*
      (BP1*(N*F=3. MEAN*F=6.2 STD*F=6.2)
       BP3*(N*F=3. MEAN*F=6.2 STD*F=6.2)
       CBP3*(N*F=3. MEAN*F=6.2 STD*F=6.2))/RTS=15;

KEYLABEL ALL='All'
        N='No.'
        MEAN='Mean'
        STD='Std Dev.';
LABEL BP1='Baseline Diastolic BP'
      BP3='Final Diastolic BP'
      CBP3='Change in Diastolic BP'
      GENDER='Gender';
FORMAT RACE $RACE. GENDER $GENDER.;

TITLE1 'TABLE 1';
TITLE2 'OUTPUT FROM PROC TABULATE';
RUN;

```

The TABULATE procedure is started with the PROC TABULATE statement. Additionally, in this example the NOSEPS option on the PROC statement is used to remove the separation lines between rows within the body of the table. If you wanted to omit all of the lines in and around the cells and table, you could use the option FORMCHAR=' ' (that's 11 blanks within the quotes).

The next statement shown in this PROC is the CLASS statement. The CLASS statement defines the classification or categorical variables that will be used in the table – in this case GENDER and RACE. The VAR statement defines the analysis variables – those for which statistics will be generated. For this example they are the beginning and ending diastolic blood pressure and the change in blood pressure – BP1, BP3, AND CBP3.

The TABLE statement tells TABULATE how you want to construct or lay out the table. Three separators are used in this example. A comma indicates moving to a new dimension, an asterisk indicates a hierarchical arrangement, and a blank space requests concatenation (side by side or above and below depending upon whether you're in the column or row dimension).

In this example there are a number of things happening in the TABLE statement. First, let's talk about the table layout. The first dimension – in this example, rows – includes the variable RACE and the keyword ALL. ALL does what it sounds like – it shows the total across the variable(s) in that dimension. Note the space between RACE and ALL. Because this is in the row dimension, the total is displayed below the last value for RACE. If the phrase had been (ALL RACE), then the total would have been shown above the first RACE value. The comma following (RACE ALL) indicates that the next variable or statistic will move to the column dimension.

In the column dimension we have asked for the GENDER class variable values to be displayed. The asterisk after GENDER indicates that the information following it should be shown within the variable(s) before the asterisk. Within each GENDER value we want to see the BP1, BP3, and CBP3 variables displayed with the number of patients, mean, and standard deviation for each of these variables. Note that statistics are grouped within parentheses for each variable and are separated by spaces, indicating that they should be displayed side by side within each variable. (With TABULATE, an asterisk should make you think 'within'.) For each statistic we also see F=X. or F=X.X. The F= is short for 'Format' and allows differing formats after the equal sign for each statistic. In this case we want N to have a format of 3 integers. For the mean and standard deviation, we have F=6.2, a format that is six columns wide with two decimal places. (When deciding on the width, don't forget to count the decimal point.)

The last thing in the TABLE statement for this example is /RTS=15. The slash is used for options on the TABLE statement, and RTS stands for 'Row Title Spacing'. The Row Titles are where the variable names or labels for rows occur. By default, they are 25% of the pagesize. RTS= allows you to control how many columns are allocated for this information.

The KEYLABEL statement is the next statement in this example. KEYLABEL has the same form and function as a LABEL statement except that it is used for TABULATE keywords, as such ALL, N, MEAN, and STD.

The FORMAT, LABEL, and TITLE statements in TABULATE function just as they do for any other PROC.

PROC REPORT

The following output demonstrates some of the capabilities of PROC REPORT.

TABLE 2 OUTPUT FROM PROC REPORT																		
Race	Gender																	
	Female						Male											
	Baseline Diastolic BP		Final Diastolic BP		Change in Diastolic BP		Baseline Diastolic BP		Final Diastolic BP		Change in Diastolic BP							
	No.	Mean	Std. Dev.	No.	Mean	Std. Dev.	No.	Mean	Std. Dev.	No.	Mean	Std. Dev.	No.	Mean	Std. Dev.	No.	Mean	Std. Dev.
===== The average change in blood pressure overall was -7.6. =====																		
White	44	94.70	5.52	44	88.23	7.14	44	-6.48	5.59	23	97.17	6.58	24	86.88	9.47	23	-10.57	6.34
Black	7	100.00	1.91	7	93.00	2.94	7	-7.00	4.32	4	98.50	5.45	4	95.75	4.65	4	-2.75	4.57
Other	1	104.00	0	1	100.00	0	1	-4.00	0	1	97.00	0	1	80.00	0	1	-17.00	0
ALL	52	95.60	5.56	52	89.10	7.00	52	-6.50	5.36	28	97.36	6.23	29	87.86	9.38	28	-9.68	6.69

The following code produced the above Table 2.

```

/**** PROC REPORT (TABLE 2) ****/

PROC REPORT DATA=PRESSURE NOWD HEADSKIP HEADLINE SPACING=1;
  COLUMN RACE (GENDER,
    (( 'Baseline/Diastolic BP' BP1 BP1=MEANBST BP1=STDBST)
    ( 'Final/Diastolic BP' BP3 BP3=MEANST BP3=STDST)
    ( 'Change in/Diastolic BP' CBP3 CBP3=CMEANST CBP3=CSTDST)));
  DEFINE RACE/GROUP ORDER=INTERNAL WIDTH=6 FORMAT=$RACE. 'Race';
  DEFINE GENDER/ACROSS FORMAT=$GENDER. SPACING=2 'Gender';
  DEFINE BP1/ANALYSIS N WIDTH=3 FORMAT=3. CENTER SPACING=3 'No.';
  DEFINE MEANBST/ANALYSIS MEAN WIDTH=8 FORMAT=8.2 CENTER 'Mean';
  DEFINE STDBST/ANALYSIS STD WIDTH=6 FORMAT=6.2 CENTER 'Std. Dev.';
  DEFINE BP3/ANALYSIS N WIDTH=3 FORMAT=3. CENTER SPACING=3 'No.';
  DEFINE MEANST/ANALYSIS MEAN WIDTH=8 FORMAT=8.2 CENTER 'Mean';
  DEFINE STDST/ANALYSIS STD WIDTH=6 FORMAT=6.2 CENTER 'Std. Dev.';
  DEFINE CBP3/ANALYSIS N WIDTH=3 FORMAT=3. CENTER SPACING=3 'No.';
  DEFINE CMEANST/ANALYSIS MEAN WIDTH=8 FORMAT=8.2 CENTER 'Mean';
  DEFINE CSTDST/ANALYSIS STD WIDTH=6 FORMAT=6.2 CENTER 'Std. Dev.';

  BREAK AFTER RACE/SKIP;

  RBREAK AFTER/ SKIP SUMMARIZE;

  COMPUTE AFTER;
    RACE='ALL';
  ENDCOMP;

  COMPUTE BEFORE;
    LINE @3 ' ';
    LINE @42 57*='';
    LINE @42 '|The average change in blood pressure overall was -7.6.|';
    LINE @42 57*='';
    LINE @3 ' ';
  ENDCOMP;

  TITLE1 'TABLE 2';
  TITLE2 'OUTPUT FROM PROC REPORT';
  RUN;

```

The REPORT procedure is started with the PROC REPORT statement. In this example we have a number of options on the PROC statement. The NOWD option indicates that you do not want to use the interactive REPORT window, but to use programming statements. By default, if you do not include the NOWD option, SAS will take you to the REPORT window where you can interactively build the report. The HEADSKIP and HEADLINE options indicate that you would like to insert a blank line (skip a line) under the column headers and underline the column headers, respectively. The SPACING= option allows you to direct REPORT in how many spaces you desire between variable columns within the report globally. You can override this within the table for specific columns and control each column individually using the SPACING= option in the DEFINE statement. This option is very useful when you are trying to fit a large amount of information into a table or want to group certain columns.

The next statement is the COLUMN statement. This is comparable to a VAR statement in PROC PRINT in that it lists what variables or columns you want included and the order in which they should be listed in the report. However, more information can be included here. In this example we want to include RACE, GENDER, and within the columns under GENDER, the number of observations, mean, and standard deviation for BP1, BP3, and CBP3. In order to include these statistics for BP1 we specify 'BP1 BP1=MEANBST BP1=STDBST'. This is called using an alias in REPORT. We will tell REPORT how to use the aliases in the DEFINE statements. This is also how you can present the same variable in more than one way. If we had wanted to see RACE displayed both as the original coded value and the formatted value, we would use an alias in the COLUMN statement and include two DEFINE statements – one for the coded value and one for the newly named column with the formatted value.

The COLUMN statement also shows headers you want to extend across columns by enclosing the header in quotes within parentheses that include the list of variables which should be grouped under the header. For this example we want the header 'Baseline Diastolic BP' to extend across BP1, MEANBST, and STDBST. The slash after the word 'Baseline' works like the SPLIT= option in PROC PRINT. It allows you to control where to divide the header.

The DEFINE statements give detailed information about each column in the report. The form of the statement is the keyword DEFINE followed by the variable or column name, a slash, and options. These options can include usage options and/or attribute options, such as column width, format, header label, spacing, and justification.

Usage options in this example include GROUP, ACROSS, and ANALYSIS. The DEFINE statement for RACE has a usage of GROUP which says that you want to summarize (collapse) observations within each level of RACE. The usage for GENDER is ACROSS which allows the table to be presented as a cross-tabulation report. For BP1, the usage is ANALYSIS, which denotes that you want to use a statistic for that variable. The N statistic specifies that you want the number of non-missing observations for BP1 within each RACE-GENDER combination. Likewise, the MEAN statistic specified for MEANBST requests that this column should be the mean for BP1 in each RACE-GENDER combination. Although the ANALYSIS usage does not need to be specified when you supply a particular statistic, it helps make the code easier to read.

Let's look more closely at the DEFINE statement for RACE and discuss the options used. After the GROUP usage option, we have specified the order in which we want the values of RACE to be displayed using the ORDER= option. By default, REPORT will order the values by formatted values. In this case, we want to retain the original order where 1=White, 2=Black, etc. Thus we specify ORDER=INTERNAL.

The WIDTH= option does exactly what it sounds like – it specifies how many spaces wide that variable column should be. The FORMAT= option allows you to assign a format to the values in the column. For this example, the RACE column has a width of six spaces and the \$RACE. format. (Note, there is no period after the width value, while format must include a period.) If there is a conflict between the specified width and format, the width option prevails.

Finally, at the end of this DEFINE statement is 'Race'. This is the header we want to see on the RACE variable column. If you don't include a header in the DEFINE statement, REPORT uses the variable's label if one is assigned in the data set.

Two other options we use on the BP1 DEFINE statement are CENTER and SPACING=. CENTER requests that the values should be center justified in the field. As mentioned earlier, the SPACING= option specifies how many spaces should be included in the report before that variable column.

The next statement we see is the BREAK statement. Here we are asking REPORT to skip a line after each value of RACE by specifying BREAK AFTER RACE/SKIP.

The RBREAK (Report Break) is used to perform some operation at the beginning or end of the report. In this case we want to skip a line and summarize the data at the bottom of the report – thus the RBREAK AFTER/SKIP SUMMARIZE.

The COMPUTE AFTER is used to include the label 'ALL' in the last row of the table. If this compute block were not included there would be no row label associated with the summary line. The ENDCOMP statement is necessary to close the COMPUTE block.

Finally, we have an additional COMPUTE block. Before the body of the table we are requesting customized information. In order to include this we specify COMPUTE BEFORE to start the customized information. The LINE statements within the block work like PUT statements in the DATA step. The LINE keyword begins a new line, the @ specifies a beginning column for the custom information, and the information within the quotes supplies the desired text or variables. So, for example, the second line statement requests that there be an equal sign repeated 57 times starting at column 42. The ENDCOMP again ends the block.

COMPUTE blocks can also be used to create new columns of information in the report. If you wanted to include a column that was the mean of BP1 and BP3, you could include a new column name in the COLUMN statement, specify the usage COMPUTED in the DEFINE statement, and include a COMPUTE block telling REPORT how to calculate the new column of information.

CONCLUSION

PROC TABULATE and PROC REPORT are very versatile procedures in Base SAS that allow tremendous flexibility in report writing. In this paper we have examined some of the capabilities of each of the procedures. Frequently, we can generate an identical report from each of the procedures. In other cases, only one of these will produce the desired output. Whether TABULATE or REPORT is better depends upon your needs and your familiarity with the PROC. If you write customized reports, I would strongly encourage you to explore both of these procedures. Many additional capabilities are available in both procedures that are not within the scope of this paper.

REFERENCES

SAS Institute Inc. (1995) *SAS Guide to the REPORT Procedure: Reference, Release 6.11*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1990) *SAS Guide to the Report Procedure: Usage and Reference, Version 6*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1990) *SAS Guide to TABULATE Processing, Second Edition*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1990) *SAS Procedures Guide, Version 6, Third Edition*, Cary, NC: SAS Institute Inc.

SAS Institute Inc. (1999) *SAS Online Doc, Version 8*, Cary, NC: SAS Institute Inc.

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

AUTHOR CONTACT INFORMATION

Debbie Buck
D. B. & P. Associates
Houston, TX
Voice: 281-256-1619
Fax: 281-256-1634
Email: debbiebuck@houston.rr.com