

It's Proc Tabulate Jim, but not as we know it!

Robert Walls, PPD, Bellshill, UK

ABSTRACT

PROC TABULATE has received a very bad press in the last few years. Most SAS® Users have come to look on it as an antiquated, cumbersome and inflexible procedure to use, when compared to PROC REPORT. What a lot of SAS® users are not aware of are the advantages of PROC TABULATE as a statistical procedure.

In this presentation we will discuss the pros and cons of using this sorely maligned procedure to analyse and manipulate data. We will present you with real life scenarios aimed to show you the strengths (and maybe a few of the weaknesses) of using PROC TABULATE (SAS® version 8.2) and how it could help to save you unnecessary coding in the future.

INTRODUCTION

This paper will not teach the basics of PROC TABULATE, but instead is designed to show some of the advantages of using this procedure and where its use could be most beneficial.

The ability to create tabulations of data is fundamental to the use of SAS® within the Pharmaceutical Industry. In fact it makes up a large amount of the work that pharmaceutical SAS® users do. PROC TABULATE, by its very nature, does just that, it tabulates data. So why has it fallen so far out of use? Is it just that people are more and more viewing it as a 'Difficult' procedure to understand? Maybe, but with the increased use and functionality of other procedures it's more likely that people just think of it as an antiquated and unattractive procedure to use.

I hope to show that PROC TABULATE still has a place in creating tabulations, indeed some of its functionalities are unique to TABULATE and can be extremely useful, especially those to do with creating counts and percentages. As well as this it can even be used to reduce the amount of code that a user needs to write in order to create output. It can still, sometimes, be tricky to make the output look as good as you'd like, though it can be used to create output data sets which can then be fed into other output procedures to get the required results. Attitudes towards PROC TABULATE should however change with the increased use of ODS, and the possibilities which it presents for the manipulation of output.

DUMMY DATA

Figure 1 below shows the dummy data set which is going to be used in all examples within this paper.

	gender	height	risk	treat	patno
1	M	64	low	A	1
2	F	87	low	B	2
3	M	55	low	C	3
4	M	84	high	C	4
5	F	84	low	C	5
6	M	87	low	D	6
7	F	78	high	D	7
8	F	69	low	D	8
9	F	52	high	E	9
10	F	66	high	F	10
11	F	88.5	high	D	11
12	F	61	low	D	12
13	F	63	high	E	13
14	F	87	low	C	14
15	F	78	low	D	15
16	M	67	low	A	16
17	M	65	high	B	17
18	M	73	low	B	18
19	M	61	high	A	19
20	M	76.5	low	E	20

Fig 1: A screen-dump showing the first twenty observations within the dummy data set used for all examples in this paper.

TABULATE BASIC CONCEPTS

PROC TABULATE provides a quick, easy and powerful way to create cross tabulations of data. It can potentially do, in a single procedure, what may otherwise take three or four procedures or data steps to do. It will, by default, create either a count or a sum of the data depending on the type of data being used (categorical or continuous, respectively), though like a lot of other procedures it has a number of other statistics which it can be used to create.

Statistics					
COLPCTN	KURTOSIS KURT	MIN	NMISS	PCTSUM	ROWPCTN
COLPCTSUM	LCLM	N	PAGEPCTN	RANGE	ROWPCTSUM
CSS	MAX	SKEWNESS SKEW	PAGEPCTSUM	REPPCTN	STDERR
CV	MEAN	STDDEV STD	PCTN	REPPCTSUM	SUM
SUMWGT	UCLM	USS	VAR	MEDIAN P50	Q1 P25
Q3 P75	P1	P5	P10	P90	P95
P99	QRANGE	PROBT	T		

Table 1: A comprehensive list of statistics that PROC TABULATE is able to create ¹. Those that are highlighted in bold are the statistical keywords associated with the creation of counts and percentages, which is what is going to be concentrated on in this paper.

CREATING PERCENTAGES

BASIC SYNTAX

Within the PROC TABULATE and RUN statements there need to be at least two statements in order to create any output. These can be either, a CLASS or a VAR statement (specifying classification or analysis variables respectively), and a table statement, which determines the layout and the tabulation of the data.

It is within the TABLE statement that the keywords must be specified in order to create the counts and percentages. For example:

```
proc tabulate data = fundata.demog(where = (gender = 'F'))
    out = work.stat;
class gender treat risk;
table gender = ' ' * treat = ' ',
    risk = ' ' * (n = 'n' pctn = '%');
run;
```

(Note: When using an analysis variable to get the percentage of the SUM which will be created the keyword PCTSUM should be used in place of the bolded PCTN in the above code).

This will give the following output:

		High Risk		Low Risk	
		n	%	n	%
Female	Drug A	4.00	12.50	5.00	15.63
	Drug B	4.00	12.50	4.00	12.50
	Placebo	.	.	2.00	6.25
	Drug A plus Drug B	2.00	6.25	6.00	18.75
	Drug A plus Placebo	2.00	6.25	.	.
	Drug B plus Placebo	1.00	3.13	2.00	6.25

Fig 2: Output created by the above code which shows a cross-tabulation of the counts of the different levels of Risk Factors, subdivided by sex and treatment (only females are presented). The denominator used for the percentages is based on a total count of the table.

Denominator: Sum of all counts = 32
 For Female, Drug A and High Risk: $(4 / 32) * 100 = 12.5\%$

It is worth noting that PROC TABULATE will automatically round the percentages, which, as with the example, can lead to them not adding up to one hundred percent. This is something which should be avoided if possible. Applying a format to three decimal places will reveal that the percentages do indeed add up to one hundred.

		High Risk		Low Risk	
		n	%	n	%
Female	Drug A	4.00	12.500	5.00	15.625
	Drug B	4.00	12.500	4.00	12.500
	Placebo	.	.	2.00	6.250
	Drug A plus Drug B	2.00	6.250	6.00	18.750
	Drug A plus Placebo	2.00	6.250	.	.
	Drug B plus Placebo	1.00	3.125	2.00	6.250

Fig 3: This is the same output as shown in Figure 2 showing the un-rounded percentage values.

USER DEFINED DENOMINATORS

The most powerful aspect of PROC TABULATE and it's ability to create counts and percentages lies in it's unique ability to allow the user to define a denominator, based on either a dimension of the table, or any of the class variables used in the tabulation. This feature is unique to tabulate and is what holds it above the other procedures which can do counts and percentages such as PROC FREQ.

DIMENSION DEFINED DENOMINATORS

The first of these methods uses a defined dimension of the table. The user specifies which dimension they wish to subdivide the table into for the creation of the denominator counts. There are four dimensions that can be specified by the user, and these are column, row, page and report, and the syntax for these is shown in Table 2.

DIMENSION DEFINED DENOMINATORS	
COLPCTN / COLPCTSUM	Uses the total of the values in the column.
ROWPCTN / ROWPCTSUM	Uses the total of the values in the row.
PAGEPCTN / PAGEPCTSUM	Uses the total of the values on the page when the page is defined as all of the output from one value of a by variable.
REPPCTN / REPPCTSUM	Uses the total of the values in the report.

Table 2: This table shows the keywords which can be used in a PROC TABULATE and the resultant dimension defined denominator used by the procedure.

These are incorporated into the PROC TABULATE code in the same manner as other percentage keywords (See previous example using PCTN).

```
proc tabulate data = fundata.demog (where = (gender = 'F'))
    out = work.stat;
class gender treat risk;
table gender = ' ' * treat = ' ',
    risk = ' ' * (n = 'n' colpctn = '%' * f = 7.3);
run;
```

		High Risk		Low Risk	
		n	%	n	%
Female	Drug A	4.00	30.769	5.00	26.316
	Drug B	4.00	30.769	4.00	21.053
	Placebo	.	.	2.00	10.526
	Drug A plus Drug B	2.00	15.385	6.00	31.579
	Drug A plus Placebo	2.00	15.385	.	.
	Drug B plus Placebo	1.00	7.692	2.00	10.526

Fig 4: This output shows the results of using the COLPCTN keyword where the sum of the counts for each column is used as the denominator for that column; hence the percentages in each column add up to one hundred.

Denominator: Sum of all Female, High Risk counts = 13
 For Female, Drug A and High Risk: $(4 / 13) * 100 = 30.769\%$

CLASS DEFINED DENOMINATORS

The second method of defining denominators is for the user to specify which of the class variables involved in the tabulation of the data should be excluded from the subdivision of the denominator counts. This will create denominator counts based on all other combinations of any other class variables.

PCTN<VAR> / PCTSUM<VAR> (VAR = a class variable name)	
	Uses the denominator counts of unique combinations of all other class variables.

Table 3: This table shows the keyword syntax for using a class defined denominator where VAR is representative of any of the class variable names.

In the following example the class variable treat will be excluded from the denominator count subdivision, and as with the other examples so far the output will be restricted to only the Female patients to simplify the output.

```
proc tabulate data = fundata.demog(where = (gender = 'F'))
    out = work.stat;
class gender treat risk;
table gender = ' ' * treat = ' ',
    risk = ' ' * (n = 'n' pctn<treat> = '%' * f = 7.3);
run;
```

		High Risk		Low Risk	
		n	%	n	%
Female	Drug A	4.00	30.769	5.00	26.316
	Drug B	4.00	30.769	4.00	21.053
	Placebo	.	.	2.00	10.526
	Drug A plus Drug B	2.00	15.385	6.00	31.579
	Drug A plus Placebo	2.00	15.385	.	.
	Drug B plus Placebo	1.00	7.692	2.00	10.526

Fig 5: Above are the results of a Female only tabulation where the denominator is taking the unique combinations of GENDER and RISK.

Denominator: Sum of all Female , High Risk counts = 13
 For Female, Drug A and High Risk: $(4 / 13) * 100 = 30.769\%$

In Figure 5 you can see that in this instance (where the output has been restricted to Females only) the overall effect is the same as that obtained through using the COLPCTN/COLPCTSUM keywords.

The output obtained when there is no restriction on the sex of the patients looks exactly the same as that above for the Females, except that when the column percentages are totalled they will sum to two hundred due to the addition of the Males

Figure 6 shows the output from the PROC TABULATE without the where clause restriction. Though each combination of risk and gender still sum to one hundred percent, the output no longer resembles that created using the COLPCTN keyword. With the addition of the male statistics, each column will add up to two hundred percent.

		High Risk		Low Risk	
		n	%	n	%
Female	Drug A	4.00	30.769	5.00	26.316
	Drug B	4.00	30.769	4.00	21.053
	Placebo	.	.	2.00	10.526
	Drug A plus Drug B	2.00	15.385	6.00	31.579
	Drug A plus Placebo	2.00	15.385	.	.
	Drug B plus Placebo	1.00	7.692	2.00	10.526
Male	Drug A	11.00	21.569	4.00	19.048
	Drug B	12.00	23.529	3.00	14.286
	Placebo	9.00	17.647	4.00	19.048
	Drug A plus Drug B	6.00	11.765	3.00	14.286
	Drug A plus Placebo	7.00	13.725	3.00	14.286
	Drug B plus Placebo	6.00	11.765	4.00	19.048

Fig 6: The results of the cross tabulation done in Figure 5 without the restriction on sex.

To replicate the ROWPCTN/ROWPCTSUM style of denominator then the RISK variable should be used within the parenthesis of the percentage keyword.

```
proc tabulate data = fundata.demog(where = (gender = 'F'))
    out = work.stat;
class gender treat risk;
table gender = ' ' * treat = ' ',
    risk = ' ' * (n = 'n' pctn<risk> = '%' * f = 7.3);
run;
```

		High Risk		Low Risk	
		n	%	n	%
Female	Drug A	4.00	44.444	5.00	55.556
	Drug B	4.00	50.000	4.00	50.000
	Placebo	.	.	2.00	100.000
	Drug A plus Drug B	2.00	25.000	6.00	75.000
	Drug A plus Placebo	2.00	100.000	.	.
	Drug B plus Placebo	1.00	33.333	2.00	66.667

Fig 7: The output obtained by excluding the RISK variable from the denominator subdivision.

Denominator: Sum of all Female, Drug A counts = 9
For Female, Drug A and High Risk: $(4 / 9) * 100 = 44.444\%$

In Figure 7 the percentage of each individual row adds up to one hundred as each row contains a unique combination of the other two class variables not specified, GENDER and TREAT.

Using the final of the three class variables, GENDER will give the sum of the percentage of every unique combination of the class variables TREAT and RISK adding to one hundred percent, as shown below:

```
proc tabulate data = fundata.demog out = work.stat;
  class gender treat risk;
  table gender = ' ' * treat = ' ',
         risk = ' ' * (n = 'n' pctn<gender> = '%' * f = 7.3);
run;
```

		High Risk		Low Risk	
		n	%	n	%
Female	Drug A	4.00	26.667	5.00	55.556
	Drug B	4.00	25.000	4.00	57.143
	Placebo	.	.	2.00	33.333
	Drug A plus Drug B	2.00	25.000	6.00	66.667
	Drug A plus Placebo	2.00	22.222	.	.
	Drug B plus Placebo	1.00	14.286	2.00	33.333
Male	Drug A	11.00	73.333	4.00	44.444
	Drug B	12.00	75.000	3.00	42.857
	Placebo	9.00	100.000	4.00	66.667
	Drug A plus Drug B	6.00	75.000	3.00	33.333
	Drug A plus Placebo	7.00	77.778	3.00	100.000
	Drug B plus Placebo	6.00	85.714	4.00	66.667

Fig 8: The results of using the class variable GENDER as the exclusion from the denominator subdivision.

Denominator: Sum of all Drug A, High Risk counts = 15
 For Female, Drug A and High Risk: $(4 / 15) * 100 = 26.667\%$

USING THE ALL KEYWORD WITH PERCENTAGES

The syntax for using the ALL keyword doesn't change with the creation of percentages, and because the ALL keyword works like a class, or analysis, variable it too can use the percentage and denominator functionalities available in PROC TABULATE.

```
proc tabulate data = fundata.demog out = work.stat;
  class gender treat risk;
  table gender = ' ' * treat = ' ' all,
         risk = ' ' *
         (n = 'n' colpctn = '%' * f = 7.3)
         all *
         (n = 'n' colpctn = '%' * f = 7.3);
run;
```

		High Risk		Low Risk		All	
		n	%	n	%	n	%
Female	Drug A	4.00	6.250	5.00	12.500	9.00	8.654
	Drug B	4.00	6.250	4.00	10.000	8.00	7.692
	Placebo	.	.	2.00	5.000	2.00	1.923
	Drug A plus Drug B	2.00	3.125	6.00	15.000	8.00	7.692
	Drug A plus Placebo	2.00	3.125	.	.	2.00	1.923
	Drug B plus Placebo	1.00	1.563	2.00	5.000	3.00	2.885
Male	Drug A	11.00	17.188	4.00	10.000	15.00	14.423
	Drug B	12.00	18.750	3.00	7.500	15.00	14.423
	Placebo	9.00	14.063	4.00	10.000	13.00	12.500
	Drug A plus Drug B	6.00	9.375	3.00	7.500	9.00	8.654
	Drug A plus Placebo	7.00	10.938	3.00	7.500	10.00	9.615
	Drug B plus Placebo	6.00	9.375	4.00	10.000	10.00	9.615
All		64.00	100.000	40.00	100.000	104.00	100.000

Fig 9: Output using the ALL keyword to create total rows and columns and the column percentage denominator keyword. The ALL row at the base of the table clearly shows the columns summing to one hundred percent.

Denominator: Sum of High Risk counts = 64
For Female, Drug A and High Risk: $(4 / 64) * 100 = 6.250\%$

Denominator: Sum of High Risk counts = 104
For Female, Drug A and ALL: $(9 / 104) * 100 = 8.654\%$

OUTPUTTING TO A DATA SET

When outputting from PROC TABULATE to an output data set, the results in the data set do not necessarily look like the output generated by the TABULATE procedure. What you see in the outputted data set are columns for the class variables, displayed in the same order in which they are specified within the class statement, three automatic variables, which contain information about the tabulation, and then a number of variables used to store each individual statistic.

```
proc tabulate data = fundata.demog(where = (gender = 'F'))
    out = work.stat;
class gender treat risk;
table gender = ' ' * treat = ' ',
        risk = ' ' * (n = 'n' pctl = '%' * f = 7.3);
run;
```

	gender	treat	risk	_TYPE_	_PAGE_	_TABLE_	N	PctN_000
1	Female	Drug A	High Risk	111	1	1	4	12.5
2	Female	Drug A	Low Risk	111	1	1	5	15.625
3	Female	Drug B	High Risk	111	1	1	4	12.5
4	Female	Drug B	Low Risk	111	1	1	4	12.5
5	Female	Placebo	Low Risk	111	1	1	2	6.25
6	Female	Drug A plus Drug B	High Risk	111	1	1	2	6.25
7	Female	Drug A plus Drug B	Low Risk	111	1	1	6	18.75
8	Female	Drug A plus Placebo	High Risk	111	1	1	2	6.25
9	Female	Drug B plus Placebo	High Risk	111	1	1	1	3.125
10	Female	Drug B plus Placebo	Low Risk	111	1	1	2	6.25

Fig 10: The output data set created from a cross tabulation of the CLASS, GENDER and TREAT variables with the count and the percentage from the overall count being created.

In the data set shown in Figure 10 it can be seen that the three class variables are recreated in the output data set, in the order specified in the class statement. Following this are the three automatic variables `_TYPE_`, `_PAGE_` and `_TABLE_` which are explained in Table 4. The TABULATE also creates a count (N) and percentage (PctN_000), the nomenclature of which will be explained in the next section.

Automatic variables created in a PROC TABULATE output data set	
<code>_TYPE_</code>	Defines the class variable combinations for each observation. This uses Boolean logic for Yes/No flags. In the above example all three class variables are tabulated in each observation so all three <code>_TYPE_</code> Boolean flags are set to 1(true).
<code>_PAGE_</code>	Contains the logical page created from using a BY variable. For each instance of a BY variable the number is incremented 1 to X.
<code>_TABLE_</code>	Represents the number of the table statement used in the TABULATE (TABULATE can tolerate numerous table statements).

Table 4: Details of the automatic variables created in a PROC TABULATE output data set.

THE OUTPUT DATA SET WITH USER DEFINED DENOMINATORS

When using user defined denominators the output data set changes slightly, because more than one type of denominator grouping can be used, more than one percentage row can be created.

```
proc tabulate data = fundata.demog out = work.stat;
  class gender treat risk;
  table gender = ' ' * treat = ' ' all,
         risk = ' ' * (n = 'n' colpctn = '%' * f = 7.3)
         all * (n = 'n' colpctn = '%' * f = 7.3);
run;
```

In the following example the total counts for the columns is used for the denominator. This means that one denominator grouping will use only the counts grouped by unique values of RISK and the other denominator grouping (because of the ALL keyword variable) will use all of the counts in the table. Hence there will be two percentage columns in the output data set to reflect this.

	gender	treat	risk	_TYPE_	_PAGE_	_TABLE_	N	PctN_001	PctN_000
13	Male	Drug B	High Risk	111	1	1	12	18.75	.
14	Male	Drug B	Low Risk	111	1	1	3	7.5	.
15	Male	Placebo	High Risk	111	1	1	9	14.0625	.
16	Male	Placebo	Low Risk	111	1	1	4	10	.
17	Male	Drug A plus Drug B	High Risk	111	1	1	6	9.375	.
18	Male	Drug A plus Drug B	Low Risk	111	1	1	3	7.5	.
19	Male	Drug A plus Placebo	High Risk	111	1	1	7	10.9375	.
20	Male	Drug A plus Placebo	Low Risk	111	1	1	3	7.5	.
21	Male	Drug B plus Placebo	High Risk	111	1	1	6	9.375	.
22	Male	Drug B plus Placebo	Low Risk	111	1	1	4	10	.
23	Female	Drug A		110	1	1	9	.	8.6538461538
24	Female	Drug B		110	1	1	8	.	7.6923076923
25	Female	Placebo		110	1	1	2	.	1.9230769231
26	Female	Drug A plus Drug B		110	1	1	8	.	7.6923076923
27	Female	Drug A plus Placebo		110	1	1	2	.	1.9230769231
28	Female	Drug B plus Placebo		110	1	1	3	.	2.8846153846
29	Male	Drug A		110	1	1	15	.	14.423076923
30	Male	Drug B		110	1	1	15	.	14.423076923
31	Male	Placebo		110	1	1	13	.	12.5
32	Male	Drug A plus Drug B		110	1	1	9	.	8.6538461538
33	Male	Drug A plus Placebo		110	1	1	10	.	9.6153846154
34	Male	Drug B plus Placebo		110	1	1	10	.	9.6153846154
35			High Risk	001	1	1	64	100	.
36			Low Risk	001	1	1	40	100	.
37				000	1	1	104	.	100

Fig 11: The data set created by the above code showing multiple percentage columns created by using different denominator groupings.

The `_TYPE_` variable in this instance has four distinct values (111, 110, 001, and 000). The class variable columns are populated to reflect the class variable combinations used in the particular observations concerned. Where `_TYPE_ = 111`, all of the class variables are populated, for `_TYPE_ = 110`, only the first two class variables (GENDER and TREAT) are populated, and for `_TYPE_ = 000` none of the class variables are populated with a value, and so on and so forth.

Figure 11 shows the output from the previous code. The two percentage columns have been created and named to reflect the denominator used within each column (Note: This does not mean that every column in the data set will use the same denominator, but rather that, within a column, the denominator is created using the same grouping of class variables).

The first column name, `PctN_001`, again employs Boolean logic as with the `_TYPE_` variable, but in this case the logic is stating which of the class variables are being used to subdivide the denominator. The Boolean flags are again representative of the class variables in the order that they appear in the class statement, so for the `PctN_001` column the only variable being used to subdivide the denominator for each observation is the RISK variable. The second column, `PctN_000`, has no subdivision of the denominator and therefore uses the total counts from the whole table as the denominator.

CONCLUSIONS

Hopefully the above examples have gone some way to prove that PROC TABULATE still has a place in the Pharmaceutical industry. If you don't like the way that the output generated by this procedure looks then you can output to a data set and display it using a procedure you consider to be easier on the eye, or else use it for creating QC output which would have the added bonus of incorporating a further level of independence to the QC.

As has been shown, PROC TABULATE can do in one easy step what it would take three or four PROC FREQ steps, or PROC SQL steps to do. Though it doesn't always look 'pretty', with the increased use of ODS there is the opportunity to manipulate the TABULATE output appearance which was not previously available.

REFERENCES

1. SAS® ONLINE DOC - <http://v8doc.sas.com/sashtml/>

CONTACT INFORMATION

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

Robert Walls
PPD
Avondale House
Phoenix Crescent
Strathclyde Business Park
Bellshill
North Lanarkshire
ML4 3NJ
+44 (0) 1698 572 632
robert.walls@eurpoe.ppd.com
www.ppd.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.