

The Utter Simplicity of the TABULATE Procedure - The Sequel

Dan Bruns, Tennessee Valley Authority, Chattanooga, TN

Not for Beginners

I had not given much thought to doing another TABULATE tutorial until a friend of mine (I won't mention any names but his initials are Frank Dilorio) asked when I was going to do "The Son of PROC TABULATE"? Well, I got to looking and found some topics that I had not really addressed in either my beginning or advanced tutorials. So this "intermediate" tutorial picks up where my beginning tutorial ("The Utter Simplicity of the TABULATE Procedure") left off, and fills a gap between the advanced tutorial. We will discover how to do subtotalling, do some rather interesting formatting, titling, and labeling, see how some of the options work and interact, and last, but of course not least, BRIEFLY discuss how the PCTN and PCTSUM with the denominator specification works. This stuff is not for the weak-of-heart, pregnant women, or anyone unfamiliar with the basics of the TABULATE procedure. The output from a CONTENTS procedure below is just so you know a little about the dataset we will be working with.

CONTENTS PROCEDURE						
Date Set Name:	SASDATA.CLASS			Observations:	27	
Member Type:	DATA			Variables:	5	
Engine:	V60x			Indexes:	0	
Created:	9:14 Wed, Sep 19			Observation Length:	48	
Last Modified:	11:40 Tue, Feb 5			Deleted Observations:	0	
Data Set Type:				Compressed:	NO	
				Reuse Space:	NO	
-----Alphabetic List of Variables and Attributes-----						
#	Variable	Type	Len	Pos	Format	Label
4	DATE	Num	8	32	DATES.	Class Date
2	LOC	Char	1	25		Location
1	NAME	Char	25	0		
3	ORG	Char	6	26		Org
5	SCORE	Num	8	40	5.1	Final Exam Score

"ALL?" the Basics

Following are a few basic examples and their totally different looking outputs by simply changing where and how the variables are coded. If these are beyond your current proficiency with TABULATE, see my Beginning Tutorials paper in the SUGI 16 and 20 proceedings and hang on to your hat because I'm starting from there and assuming you understand this much.

```
PROC TABULATE DATA=CLASS FORMAT=5.1;
  CLASS ORG LOC DATE;
  VAR SCORE;
  TABLE ORG, LOC*SCORE*(N MEAN);
```

	Location					
	A		B		C	
	Final Exam Score		Final Exam Score		Final Exam Score	
	N	MEAN	N	MEAN	N	MEAN
Org						
Energy	4.0	84.4	2.0	96.4	.	.
Mgt S	2.0	73.4	1.0	85.4	7.0	84.8
Power	4.0	89.0	2.0	70.7	3.0	79.8

In the above example you see a complete PROC step with a column for the count(N) and mean of SCORE for each location.

TABLE ORG*LOC,
SCORE*(N MEAN MAX PCTN);

		Final Exam Score			
		N	MEAN	MAX	PCTN
Org	Location				
Energy	A	4.0	84.4	93.0	16.0
	B	2.0	96.4	100.0	8.0
Mgt S	A	2.0	73.4	99.4	8.0
	B	1.0	85.4	85.4	4.0
	C	7.0	84.8	98.3	28.0
Power	A	4.0	89.0	99.1	16.0
	B	2.0	70.7	98.0	8.0
	C	3.0	79.8	93.6	12.0

These are the same numbers from the previous output. Location was simply moved from the column expression to the row expression.

TABLE ORG LOC,
SCORE*(N*F=5.0 MEAN MAX PCTN);

		Final Exam Score			
		N	MEAN	MAX	PCTN
Org					
Energy		6	88.4	100.0	24.0
Mgt S		10	82.6	99.4	40.0
Power		9	81.9	99.1	36.0
Location					
A		10	84.0	99.4	40.0
B		5	83.9	100.0	20.0
C		10	83.3	98.3	40.0

Double your productivity with two tables in one: the N, MEAN, MAX, and PCTN statistics in the column expression allows you to use the row expression to see a summary by two different variables (ORG and LOC) in one table.

TABLE ORG ALL,
(LOC ALL)*SCORE*(N*F=5.0 MEAN);

	Location							
	A		B		C		ALL	
	Final Exam Score		Final Exam Score		Final Exam Score		Final Exam Score	
	N	MEAN	N	MEAN	N	MEAN	N	MEAN
Org								
Energy	4	84.4	2	96.4	.	.	6	88.4
Mgt S	2	73.4	1	85.4	7	84.8	10	82.6
Power	4	89.0	2	70.7	3	79.8	9	81.9
ALL	10	84.0	5	83.9	10	83.3	25	83.7

What in the world happened in this last example? There's no variable named ALL in the dataset? That's right, but ALL is kind of like a builtin class variable that can be specified to accumulate totals for the entire row and/or column. In the above example it was used in the row expression to produce a set of totals after the ORG rows-which totals the columns. The use of ALL in the column expression caused it to produce a column after the LOC columns-which totals the rows. The fact that you put the ALL in the other expression requires you to remember what the TABLE expression is for: to define what you want and where and how. Remember, ORG ALL above says to give us a set of rows for each ORG value and concatenate the ALL row. Also notice since ALL is grouped with LOC and then nested, the column contains the totals for all locations using the same statistics.

If you placed the ALL before the ORG variable (i.e. ALL ORG) you would get the totals as the first row of the table as below.

TABLE ALL ORG,
ALL*SCORE*(N*F=4.0 MEAN MIN MAX)
LOC*SCORE*(N*F=4.0 MEAN)
/ RTS=12 ;

	Location							
	ALL				A		B	
	Final Exam Score				Final Exam Score		Final Exam Score	
	N	MEAN	MIN	MAX	N	MEAN	N	MEAN
ALL	25	83.7	47.3	100.0	10	84.0	5	83.9
Org								
Energy	6	88.4	69.9	100.0	4	84.4	2	96.4
Mgt S	10	82.6	47.3	99.4	2	73.4	1	85.4
Power	9	81.9	51.3	99.1	4	89.0	2	70.7

You may not be able to tell from these examples, but TABULATE computes true statistics (i.e. MEAN above). That means it DOES NOT add-up the means from the tables and then divide by the number of tables entries; it accumulates each observations value and divides by the number of observations with non-missing values.

Titles and Labels

You can see that to have TABULATE put descriptive titles or labels for the variables you simply need to assign meaningful labels to them. You can either do this in earlier steps that create the dataset or with a LABEL statement in the PROC step. But what about the statistics and ALL? Simply attach a descriptive label to any variable or statistic right in the TABLE statement. Follow it with an equals sign(=) and a quoted label ('This is a

Label') just like you do in a LABEL statement. Or if you want to use a certain label for every use of the statistic, use the KEYLABEL statement which looks exactly like the LABEL statement except you use the statistic's name instead of a variable name. Following is an example of doing both.

KEYLABEL N='Count'
MEAN='Avg'
ALL='Total';
TABLE ORG="Department" ALL,
(LOC ALL='Row Totals')*SCORE*(N*F=5.0 MEAN)
/ RTS=12 BOX='SESUG 95';

SESUG 95	Location							
	A		B		C		Row Totals	
	Final Exam Score		Final Exam Score		Final Exam Score		Final Exam Score	
	Count	Avg	Count	Avg	Count	Avg	Count	Avg
Department								
Energy	4	84.4	2	96.4	.	.	6	88.4
Mgt S	2	73.4	1	85.4	7	84.8	10	82.6
Power	4	89.0	2	70.7	3	79.8	9	81.9
Total	10	84.0	5	83.9	10	83.3	25	83.7

The above example has another tables option specified (BOX=) that specifies what to put in the upper-left corner box of the table. You can specify a quoted string as above or a variable as in the next example.

In the following example we added the MISSING and NOSEPS options to the PROC statement to have TABULATE treat missing values as a valid category (which it does not do by default) and remove the separation lines between the rows. I also specified some table options: BOX=SCORE to label the upper-left box with the SCORE variable's label; and MISSTEXT='None' to label missing values in the tables with the text 'None' instead of the standard period.

PROC TABULATE DATA=CLASS MISSING NOSEPS ;
CLASS ORG LOC DATE;
VAR SCORE;
LABEL ORG="Department"
SCORE="Final Exam Averages";
KEYLABEL N='Count'
MEAN='Avg'
ALL='Total';
TABLE ORG ALL='-- Totals --',
(LOC ALL='Row Totals')
***(SCORE*MEAN='*F=5.1)**
/BOX=SCORE ROW=FLOAT MISSTEXT='None';

Final Exam Averages	Location							
					A		B	
					Final Exam		Final Exam	
					Aver-		Aver-	
Department								
Energy	None	84.4	96.4	None	88.4			
Mgt S	None	73.4	85.4	84.8	82.6			
Power	None	89.0	70.7	79.8	81.9			
-- Totals --	None	84.0	83.9	83.3	83.7			

Notice that since the MEAN label was blank and the ROW=FLOAT was specified, that no space is wasted for it. Basically ROW=FLOAT removes rows in the table that are blank and "floats" the rest of the lines up. Also notice the additional row and column for the "blank" (missing) organization and location that has appeared because of the MISSING option on

the PROC statement. Even though the MISSING options created the column the "None" indicates the scores are missing.

The label "Final Exam Averages" appears several times and seems unnecessary, so let's get rid of it out of the column label and make a table that doesn't look like a table.

```
PROC TABULATE DATA=CLASS MISSING NOSEPS
  FORMCHAR=' ';
  CLASS ORG LOC DATE;
  VAR SCORE;
  LABEL ORG="Department"
  SCORE="Final Exam Averages";
  KEYLABEL N="Count"
  MEAN="Avg"
  ALL="Total";
  TABLE ORG ALL='--- Totals ---',
    (LOC ALL='Dept Totals')
    *(SCORE=' '*MEAN=' '*F=6.1)
  / BOX=SCORE ROW=FLOAT MISSTEXT='None';
```

Final Exam Averages	Location				Dept Totals
	A	B	C		
Department	None	None	None	None	None
Energy	None	84.4	96.4	None	88.4
Mgt S	None	73.4	85.4	84.8	82.6
Power	None	89.0	70.7	79.8	81.9
--- Totals ---	None	84.0	83.9	83.3	83.7

By simply adding the FORMCHAR= option to the PROC statement and specifying 16 blanks, you change all the characters that TABULATE uses to "draw" the box or frame to blanks. If you have access to a laser printer you can also use characters that form "solid" lines around your table. And like in any other SAS procedure, you can use user-defined formats to enhance the output.

```
PROC FORMAT;
  value $locs
    "A"="Addams" "B"="Beetlejuice"
    "C"="Carmen" other="????";
PROC TABULATE DATA=CLASS MISSING NOSEPS
  FORMCHAR='TABFACCCBCEB8FECABCB
  BB4E7E4F60AFE04C6E40X'; /* for IBM 3820 */
  CLASS ORG LOC DATE;
  VAR SCORE;
  FORMAT LOC $locs.;
  LABEL ORG="Department"
  SCORE="Final Exam Averages";
  KEYLABEL N="Count"
  MEAN="Avg"
  ALL="Total";
  TABLE ORG ALL='--- Totals ---',
    (LOC ALL='Department Totals'*F=10.1)
    *(SCORE=' '*MEAN=' '*F=7.1)
  / RTS=20 BOX=SCORE
  ROW=FLOAT MISSTEXT='None';
```

Final Exam Averages	Location				Department Totals
	????	Addams	Beetle- juice	Carmen	
Department	None	None	None	None	None
Energy	None	84.4	96.4	None	88.4
Mgt S	None	73.4	85.4	84.8	82.6
Power	None	89.0	70.7	79.8	81.9
--- Totals ---	None	84.0	83.9	83.3	83.7

The combination of these options has proved to be a real challenge, but can produce some very interesting reports.

```
PROC TABULATE DATA=CLASS MISSING FORMAT=6.1;
  CLASS ORG LOC DATE;
  VAR SCORE;
  LABEL ORG="Department"
  SCORE="Final Exam Averages";
  KEYLABEL N="Count"
  MEAN="Avg"
  ALL="Total";
  TABLE DATE*SCORE='
    *(MEAN=' '*MAX=' '*MIN=' ')?,
  LOC
  / RTS=15 ROW=FLOAT
  BOX="Final Exam: Avg/Max/Min" ;
```

Final Exam: Avg/Max/Min	Location			
	A	B	C	
Class Date				
	.	.	.	.
	.	.	.	.
	.	.	.	.
07APR	.	99.3	70.7	.
	.	99.4	90.0	.
	.	99.1	51.3	.
03MAY	.	91.2	89.1	.
	.	93.5	92.8	.
	.	88.9	85.4	.
22JUN	.	70.0	.	84.1
	.	70.1	.	98.3
	.	69.9	.	64.6
12OCT	.	79.9	100.0	81.4
	.	93.5	100.0	90.8
	.	47.3	100.0	66.5

Can you tell what the three lines for each date represent? Only the label in the box gives us a clue.

```
TABLE SCORE='*(MEAN=' '*MAX=' '*MIN=' ')?,
  DATE="Final Exam: Avg/Max/Min"
  / RTS=15 ROW=FLOAT;
```

Final Exam: Avg/Max/Min				
	07APR	03MAY	22JUN	12OCT
.	85.0	90.1	81.0	83.0
.	99.4	93.5	98.3	100.0
.	51.3	85.4	64.6	47.3

This one is rather interesting, but I haven't found a way to get rid of the row titles. But you can "hide" them by using the FORMCHAR=' ' and NOSEPS options.

Final Exam: Avg/Max/Min				
	07APR	03MAY	22JUN	12OCT
.	85.0	90.1	81.0	83.0
.	99.4	93.5	98.3	100.0
.	51.3	85.4	64.6	47.3

Advanced Tutorials

Or you can just use a label in the row title.

```
TABLE SCORE='*(MEAN MAX MIN)*F=5.1,
DATE='* LOC='*
/ RTS=12 ROW=FLOAT BOX='Final Exam Scores';
```

Final Exam		07APR	03MAY	22JUN	12OCT		A	B
Avg		85.0	90.1	81.0	83.0		84.0	83.9
MAX		99.4	93.5	98.3	100.0		99.4	100.0
MIN		51.3	85.4	64.6	47.3		47.3	51.3

Subtotalling

The only real trick to doing subtotalling is the nesting of ALL in the row expression.

```
TABLE ORG*(LOC ALL='**Subtotal') ALL='Dept Total',
SCORE='*MEAN*F=6.1
/ RTS=25 BOX=SCORE
ROW=FLOAT MISSTEXT='None';
```

Final Exam Averages			Avg
Department	Location		
	A		None
	**Subtotal		None
Energy	Location		
			None
	A		84.4
	B		96.4
	**Subtotal		88.4
Mgt S	Location		
	A		73.4
	B		85.4
	C		84.8
	**Subtotal		82.6
Power	Location		
	A		89.0
	B		70.7
	C		79.8
	**Subtotal		81.9
Dept Total			83.7

Above we see the nesting of (LOC ALL) in ORG. This tells TABULATE to concatenate an ALL row after all the LOC rows for each ORG value.

Here is a table that I wasn't sure could be done!

```
TABLE ORG*(LOC DATE ALL='**Subtotal')
ALL='All Departments',
SCORE='*MEAN*F=6.1
/ RTS=25 BOX=SCORE
ROW=FLOAT MISSTEXT='None';
```

Final Exam Averages		Avg
Department	Location	
	A	None
	Class Date	None
	**Subtotal	None
Energy	Location	
	A	84.4
	B	96.4
	Class Date	
	07APR	None
	03MAY	90.9
	22JUN	69.9
	12OCT	92.9
	**Subtotal	88.4
Mgt S	Location	
	A	73.4
	B	85.4
	C	84.8
	Class Date	
	07APR	99.4
	03MAY	85.4
	22JUN	87.3
	12OCT	72.9
	**Subtotal	82.6
Power	Location	
	A	89.0
	B	70.7
	C	79.8
	Class Date	
	07APR	80.1
	03MAY	93.5
	22JUN	77.4
	12OCT	93.5
	**Subtotal	81.9
All Departments		83.7

The same table as earlier, but we have added DATE to the grouping for subtotalling. The computation of the subtotal is not affected by the additional rows for DATE since all it has to total is the MEAN of SCORE.

Formatting

Just about anywhere you can specify a format (FORMAT= on the PROC statement or F= in the table definitions) you can use a SAS output format, i.e. DOLLAR12.2, COMMA10.0, etc., or a user-defined format.

```
PROC FORMAT;
VALUE GRADE
LOW-60="F"
60<-70="D"
70<-80="C"
80<-90="B"
90<-HIGH="A";
PROC TABULATE DATA=CLASS FORMAT=2.0;
CLASS ORG LOC DATE SCORE;
FORMAT SCORE GRADE;
KEYLABEL N='';
TABLE ORG ALL, (LOC ALL)*SCORE="Grade"
/ RTS=10 ROW=FLOAT;
```

	Location												ALL							
	A					B					C									
	Grade					Grade					Grade					Grade				
	F	D	C	B	A	F	D	C	B	A	F	D	C	B	A	F	D	C	B	A
Org																				
Energy			1		2	1			2								1		2	3
Mgt S	1	1			1	1	1		1	2	1	3	1	1	2	2	4			
Power			1	1	3	1	1		1		1	1	1	1	1	1	1	2	4	
ALL	1	1	1	1	2	5	1	2	2	2	2	4	2	3	3	6	11			

The only real problem is the grades are backwards. To tell TABULATE to use the formatted values in ordering the columns, use the ORDER=FORMATTED option.

```
PROC TABULATE DATA=CLASS FORMAT=2.0
  ORDER=FORMATTED;
  CLASS ORG LOC DATE SCORE;
  FORMAT SCORE GRADE;
  KEYLABEL N='';
  TABLE ORG ALL, (LOC ALL)*SCORE="Grade"
  / RTS=10 ROW=FLOAT;
```

	Location															ALL	
	A					B					C						
	Grade					Grade					Grade						
	A	B	C	D	F	A	B	C	D	F	A	B	C	D	F		
Org																	
Energy	1	2		1		2							3	2		1	
Mgt S	1	1				1		1		3	1	2	1	4	2	2	1
Power	3		1				1	1	1	1		1	4	2	1	1	1
ALL	5	2	1	1	1	2	2	1	4	2	2	2	11	6	3	3	2

Or another way to look at it may be ...

```
PROC FORMAT;
  VALUE GRADE
    LOW-60=" F "
    60<-70=" D "
    70<-80=" C "
    80<-90=" B "
    90<-HIGH=" A ";
  PROC TABULATE DATA=CLASS MISSING NOSEPS
    FORMCHAR='';
  CLASS ORG LOC DATE;
  VAR SCORE;
  FORMAT LOC SLOCS.;
  LABEL ORG="Department"
    SCORE="Final Exam Averages";
  KEYLABEL N="Count"
    MEAN="Avg"
    ALL="Total";
  TABLE ORG ALL="-- Totals --",
    (LOC ALL="Dept Totals")
    *(SCORE="--MEAN="--F=grade.)
  / BOX=SCORE ROW=FLOAT MISSTEXT="None";
```

Final Exam Averages	Location				
	Beet- Adda- ms	leju- ice	Carm- en	Tota- ls	Dept
Department	None	None	None	None	None
Energy	None	B	A	None	B
Mgt S	None	C	B	B	B
Power	None	B	C	C	B
--- Totals ---	None	B	B	B	B

But this looks rather ridiculous, so simply give the format some width and "POOF", it's fixed.

```
TABLE ORG DATE ALL="-- Totals --",
  (LOC="-- ALL="Totals")
  *(SCORE="--MEAN="--F=grade7.)
  / BOX=SCORE ROW=FLOAT MISSTEXT="--None--";
```

Final Exam Average Grades	????	Addms	Beetle- juice	Carmen	Totals
Department	None	None	None	None	None
Energy	None	B	A	None	B
Mgt S	None	C	B	B	B
Power	None	B	C	C	B
Class Date	None	None	None	None	None
07APR	None	A	C	None	B
03MAY	None	A	B	None	A
22JUN	None	C	None	B	B
12OCT	None	C	A	B	B
--- Totals ---	None	B	B	B	B

This question always seems to come up: "How do I know where to put the format specification? In a FORMAT statement or a F= option?"

Here is a handy rule-of-thumb:

Use a FORMAT statement to format a CLASS variable; and use a format option (F=) to format an ANALYSIS variable.

Percentages

In its simplest form the PCTN or PCTSUM is just another statistic like N or MEAN you can request.

```
PROC TABULATE DATA=CLASS FORMAT=6.1;
  CLASS ORG LOC DATE;
  VAR SCORE;
  TABLE ORG, (LOC ALL)*(N*F=3.0 PCTN);
```

	Location							
	A		B		C		ALL	
	N	PCTN	N	PCTN	N	PCTN	N	PCTN
Org								
Energy	6	22.2	2	7.4	.	.	8	29.6
Mgt S	2	7.4	1	3.7	7	25.9	10	37.0
Power	4	14.8	2	7.4	3	11.1	9	33.3

Unless otherwise specified, the percentage is computed based on all the observations in the dataset. Notice that the PCTN under the ALL column does not add up to 100 due to rounding.

To specify how the percentage is computed you simply attach a denominator specification to PCTN or PCTSUM using the inequality signs less-than (<) and greater-than (>). The real trick to understanding how the denominator specification works is to remember you are telling TABULATE what the denominator is to divide into the N or SUM value.

```
TABLE ORG, (LOC ALL)*(N*F=3.0 PCTN < ORG >);
```

	Location							
	A		B		C		ALL	
	N	PCTN	N	PCTN	N	PCTN	N	PCTN
Org								
Energy	6	50.0	2	40.0	.	.	8	29.6
Mgt S	2	16.7	1	20.0	7	70.0	10	37.0
Power	4	33.3	2	40.0	3	30.0	9	33.3

The above example shows the row expression in the denominator specification. Notice that none of the counts(N) have changed but the PCTN values have because the denominator has changed from the entire dataset (27 observations) to all the observations for ORG within that columns(LOC) value. Observe that since PCTN is nested in LOC that the denominator specification is saying to divide each cell under that location by the total number of observations that are in that location. So why do you specify the row expression? Because that is simply telling TABULATE which number of observations to total. So, in the above example, we see that location A cells are divided by 12, the total of all the ORG observations in that location. For location B we see each cell is divided by 5, the total of all the ORG observations in that location. And for location C we see each cell is divided by 10, the total of all the ORG observations in that location. And for the ALL column we see each cell is divided by the total of all the ORG observations in all the locations.

Here is a handy rule-of-thumb:

To get percentages by column, use the row expression; and to get percentages by row, use the column expression.

TABLE ORG, (LOC ALL)*(N*F=3.0 PCTN < LOC ALL >);

	Location							
	A		B		C		ALL	
	N	PCTN	N	PCTN	N	PCTN	N	PCTN
Org								
Energy	6	75.0	2	25.0	.	.	8	100.0
Mgt S	2	20.0	1	10.0	7	70.0	10	100.0
Power	4	44.4	2	22.2	3	33.3	9	100.0

Notice in the above example that the entire column expression is coded as the denominator specification. If you don't, strange results or even errors can occur. As before, you are simply telling TABULATE which number of observations to total. So, in the above example, we see that organization 'Energy' cells are divided by 8, the total of all the LOC observations in that organization. For organization 'Mgt S' we see each cell is divided by 10, the total of all the LOC observations in that organization. For organization 'Power' we see each cell is divided by 9, the total of all the LOC observations in that organization. And for the ALL column we see each cell is divided by the total of all the LOC observations in that organization, thus the 100 percent.

I'll leave you with the same example as above except using PCTSUM instead of PCTN, just so you can see it works the same way.

TABLE ORG,
(LOC ALL)*SCORE
*(SUM*F=5.1 PCTSUM < LOC ALL >);

	Location							
	A		B		C		ALL	
	Final Exam Score		Final Exam Score		Final Exam Score		Final Exam Score	
	SUM	PCTSUM	SUM	PCTSUM	SUM	PCTSUM	SUM	PCTSUM
Org								
Energy	337.6	63.7	192.8	36.3	.	.	530.4	100.0
Mgt S	146.7	17.8	85.4	10.3	593.5	71.9	825.6	100.0
Power	356.2	48.3	141.3	19.2	239.4	32.5	736.9	100.0

This is hardly BRIEF as promised, but hopefully enough to get you started using percentages. The more complicated your "crossings", as the SAS manuals refer to them, the tougher it is going to be to determine the denominator specification.

In Summary

This paper is not intended to be a cure for all your TABULATE problems. Every use of TABULATE is unique in some ways. All I have attempted to do is give you a good starting point or foundation to better understand how to get TABULATE to give you what you want.

So good luck and happy tabulating!!!

Acknowledgements

For a complete discussion of TABULATE and its uses see the "SAS Guide to TABULATE Processing, Second Edition" and the "SAS Language and Procedures, Usage, Version 6, First Edition", chapter 25, "Creating Summary Tables". There is a very good article in the first issue of "Observations, The Technical Journal for SAS Software Users", vol. 1, no. 1, entitled "Computing Percentages with PROC TABULATE" by Tina Keene.

SAS is a registered trademark of the SAS Institute, Inc., Cary, NC.

Author

If you have any questions or comments, please write or call:

Dan Bruns
Tennessee Valley Authority
1101 Market Street(MP 2B)
Chattanooga, TN 37402
(615) 751-6430 FAX: (615) 751-3163
EMail: spdeh@mhs-tva.attmail.com