SESUG Paper 199-2019

Using PROC TABULATE and ODS Style Options to Make Really Great Tables

Wendi L. Wright, Questar Assessments

Abstract

We start with an introduction to PROC TABULATE, looking at the basic syntax, and then building on this syntax by using examples. Examples will show how to produce one-, two-, and three-dimensional tables using the TABLE statement. More examples cover how to choose statistics for the table, labeling variables and statistics, how to add totals and subtotals, working with percents and missing data, and how to clean up the table using options. A look at the three ways you can use the PRELOADFMT option is also covered.

Next the ways to use the ODS STYLE= option in PROC TABULATE are covered. This option helps to customize the tables and improve their attractiveness. This option is very versatile and, depending on where the option is used, can justify cells or row and column headings, change colors for both the foreground and background of the table, modify lines and borders, controlling cell width, add a flyover text box in ODS HTML, or add GIF figures to the row or column headings.
his paragraph uses the PaperBody style.

Introduction

PROC TABULATE is a procedure that displays descriptive statistics in tabular format. It computes many statistics that other procedures compute, such as MEANS, FREQ, and REPORT and displays these statistics in a table format. TABULATE will produce tables in up to three dimensions and allows, within each dimension, multiple variables to be reported one after another hierarchically. There are also some very nice mechanisms that can be used to label and format the variables and the statistics produced.

BASIC SYNTAX

```
PROC TABULATE <options>;
    CLASS variables < / options>;
    VAR variables < / options>;
    TABLE  <page> ,
           <row> ,
           column
           < / options> ;
    … other statements … ;
RUN;
```

Let's take a look at the basic syntax of the PROC TABULATE Procedure. We will start with three of the statements that you can use in PROC TABULATE, CLASS, VAR, and TABLE. As you can see each of these statements, as well as the PROC TABULATE statement itself allows options to be added. For each of the statements, the options need to be preceded with a '/'.

Note two differences in the syntax from any other Procedure in SAS®. One, the variables in all three statements can not be separated by commas. And two, the commas in the table statement are treated in a special way and mean a change in dimension.

VAR Statement

The VAR statement is used to list the variables you intend to use to create summary statistics on. As such, they must be numeric.

CLASS Statement

Classification variables allow you to get statistics by category. You will get one column or row for each value of the CLASS variable. You will need make sure you use a categorical variable with a limited number of categories or you may end up producing more pages of output than you intended.

The syntax for the CLASS statement is similar to the VAR statement. You list the variables you want to use for grouping data followed by a '/' and any options you want. The variables here can be either numeric or character (unlike the VAR statement which requires numeric). The statistics you can request for these variables are only counts and percents. The statistics will be produced for each LEVEL of the variable. This is almost like using a BY statement within the table.

TABLE statement

The Table statement consists of up to three dimension expressions and the table options. To identify different dimensions, use commas. The order of the dimensions is page, row, and column. If you only specify one dimension, then it is assumed to be column. If two are specified, row, then column. Options appear at the end after a '/'.

You can have multiple table statements in one PROC TABULATE. This will generate one table for each statement. All variables listed in the table statement must also be listed in either the VAR or CLASS statements. In the table expressions, there are many statistics that can be specified. Among them are row and column percents, counts, means, and percentiles.

CONSTRUCTING A TABLE STATEMENT – DIMENSION EXPRESSIONS

There are many elements you can use to construct a table expression. You start with the variables you want to include in the table, but you can also specify the universal CLASS variable ALL which allows you to get totals. You will also need to specify what statistics you want to put in the cells of the table. To make your table 'pretty', you can also specify formats, labels, and ODS style specifications in the expression.

We begin by taking a closer look at constructing dimension expressions. Here is where PROC TABULATE differs from all the other Procedures in the SAS programming language. The syntax used for PROC TABULATE is very different.

- A comma specifies to add a new dimension.
- The asterisk is used to produce a cross tabulation of one variable with another (within the same dimension however, different from PROC FREQ).
- A blank is used to represent concatenation (i.e. place this output element after the preceding variable listed).
- Parenthesis will group elements and associate an operator with each element in the group
- Angle brackets specify a denominator definition for use in percentage calculations.

Constructing a Single Dimensional Table

The code below will produce a single dimension table. We have one VAR variable – income, and one CLASS variable – gender. To request what statistic you would like to see, use a '*' and add the name of the statistic next to the VAR variable you want to use for producing the statistics. As demonstrated below, you can group multiple statistics and variables with parentheses to get the results you want. We are also requesting that mean income be produced for each level of the class variable gender.

Notice that there are NO commas included in the TABLES statement. This indicates to SAS that this is to be a one dimensional table.

```
PROC TABULATE data=one;
      CLASS GENDER;
      VAR income;
      TABLE income * (N Mean)   INCOME * MEAN * GENDER;
RUN;
```

Income		Income	
		Mean	
Income		Gender	
N	Mean	Female	Male
30.00	52111.80	52000.69	52238.79

Adding Statistics

The code above requested two statistics, N and Mean. You can produce many other statistics. Below is a table of them. If you do not provide a statistic name, the default statistic produced will be 'N' for the CLASS variables and 'SUM' for the VAR variables.

Descriptive Statistics	Quantile Statistics
COLPCTN	MEDIAN | P50
PCTSUM	P1
COLPCTSUM	Q3 | P75
MAX	P90
ROWPCTN	P95
MEAN	P5
ROWPCTSUM	P10
MIN	P99
STDDEV / STD	Q1 | P25
N	QRANGE
STDERR	
NMISS	
SUM	Hypothesis Testing
PAGEPCTSUM	
PCTN	ProbT
VAR	T

Two Dimensional Table

To create a two dimensional table, we need to add some additional detail to our TABLES statement. We specify first the row dimension, then the column dimension, separated by a comma. You can get very different table structures by changing where the statistic definitions are placed. They can be attached to either the VAR or the CLASS variable, but the numbers in the cells will ALWAYS be calculated using the VAR variable(s).

Here we have the statistics attached to the columns.

```
PROC TABULATE data=one;
      CLASS gender;
      VAR income;
      TABLE gender,
            income * (N Mean Max) ;
RUN;
```

	Income		
	N	Mean	Max
Gender			
Female	16.00	52000.69	93849.00
Male	14.00	52238.79	78695.00

If you move the statistic specification so that it is attached to the rows, the results look very different.

```
PROC TABULATE data=one;
      CLASS gender;
      VAR income;
      TABLE gender * (N Mean Max) ,
            income ;
RUN;
```

		Income
Gender		
Female	N	16.00
	Mean	52000.69
	Max	93849.00
Male	N	14.00
	Mean	52238.79
	Max	78695.00

More than One Classification Variable

You can specify multiple classification variables. They can be used in any of the dimensions and can be nested. In our example, we have three CLASS variables. Two of the CLASS variables are nested in the row dimension, Fulltime * Gender.

When you have multiple CLASS variables, I recommend using the option MISSING. By default, The CLASS statement tells SAS to drop observations that have a missing value in even ONE of the variables. If you specify the MISSING option in your CLASS statement, SAS does NOT remove that observation from all the tables and will consider the missing values as valid levels for the CLASS variable(s).

```
PROC TABULATE data=one;
      CLASS gender fulltime educ / MISSING;
      VAR income;
      TABLE fulltime * gender ,
            Income * educ * mean ;
RUN;
```

		Income			
		Educ			
		High School	Bachelors	Masters	Doctorate
		Mean	Mean	Mean	Mean
Fulltime	Gender				
Fulltime	Female		42771.20		74589.60
	Male	52506.67		50729.25	56466.00
Parttime	Female	44788.67	36947.00		
	Male	35600.00		62258.00	

Adding Totals and Subtotals

In order to get marginal statistics in your table, you use the 'ALL' keyword. You can use the keyword in multiple places and, depending on where you put the keyword, there will be different subtotals produced. You can place the keyword on the row or the column dimensions or on both. In this example we are requesting subtotals for Gender and overall totals for both the row and the column dimensions.

```
PROC TABULATE data=one;
      CLASS gender fulltime educ;
      TABLE (fulltime ALL) * gender ALL,
            educ * N;
RUN;
```

		Educ			
		High School	Bachelors	Masters	Doctorate
		N	N	N	N
Fulltime	Gender				
Fulltime	Female		5.00		5.00
	Male	3.00	1.00	4.00	2.00
Parttime	Female	3.00	4.00	2.00	
	Male	3.00	1.00	3.00	
All	Female	3.00	9.00	2.00	5.00
	Male	6.00	2.00	7.00	2.00
All		9.00	11.00	9.00	7.00

A few additional code examples are here. These demonstrate that the ALL keyword can be used as many times as you wish.

```
PROC TABULATE data=one;
      CLASS gender fulltime educ;
      TABLE fulltime * (gender ALL) ,
            (educ all)* N ;
RUN;

PROC TABULATE data=one;
      CLASS gender fulltime educ;
      TABLE (fulltime ALL) * (gender ALL) * ALL ,
            (educ all)* N ;
RUN;
```

Adding and Hiding Row and Column Labels

There are two ways to add labels for your variables. The first is the simplest. Just add the text ='label' after the variable name to the TABLE dimension expression. This will work for both variables and statistics. The second way is to add a LABEL statement for variables and/or a KEYLABEL statement for statistics to your code. For example:

LABEL var='label';
KEYLABEL stat='label';

In order to hide variable or statistic labels, you leave the label specification blank (i.e. =' ').

The following example demonstrates the simpler way to add and hide labels.

```
PROC TABULATE data=one;
      CLASS educ gender fulltime;
      VAR income;
      TABLE educ ,
            Fulltime='Employment Status' * gender = '  '  *
                  income * mean = ' ' ;
RUN;
```

	Employment Status			
	Fulltime		Parttime	
	Female	Male	Female	Male
	Income	Income	Income	Income
Educ				
High School		52506.67	44788.67	35600.00
Bachelors	42771.20		36947.00	
Masters		50729.25		62258.00
Doctorate	74589.60	56466.00		

Alternatively, you can use this code to produce the table above:

```
PROC TABULATE data=one;
      CLASS educ gender fulltime;
      VAR income;
      TABLE educ,
            Fulltime * gender * income * mean ;
      LABEL gender=' ' Fulltime='Employment Status';
      KEYLABEL mean=' ';
RUN;
```

Filling the Big White Box

To fill in the big white box in the upper left, use the BOX= option.

```
PROC TABULATE data=one;
      CLASS gender fulltime;
      VAR income;
      TABLE fulltime = 'Employment Status',
            Gender * income * mean
            / BOX='Mean Income' ;
RUN;
```

Mean Income	Gender	
	Female	Male
	Income	Income
	Mean	Mean
Employment Status		
Fulltime	58680.40	52596.56
Parttime	40867.83	51594.80

Three Dimensional Tables

Three dimensional tables are easy to produce, just add another section BEFORE the row and column expressions in the table statement. PROC TABULATE will now interpret the dimension statements in this order, first page, then the row, then the columns.

Three dimensional tables have a nice way to fill in the upper left area. Instead of the label of the page dimension appearing above the table, you can use the BOX=_page_ option to place that label inside the big white box. Only a part of the output is included here.

```
PROC TABULATE data=one;
      CLASS gender fulltime educ;
      VAR income;
      TABLE educ='Education',
            fulltime = 'Employment Status',
            Gender * income * mean
            / BOX=_PAGE_ ;
RUN;
```

This will produce a separate table for each level of education with the value of the variable education in the big white box.

Producing Cell Percents

One of the nice things about PROC TABULATE is that it allows you to put both percents and summary statistics in the same table. You can get percentages on both numeric and character variables as the percents are based only on counts. The VAR statement is not needed unless you want to add other summary statistics besides count and percents. Just use the CLASS statement. Several types of percents can be requested or you can construct your own percentages by specifying the denominator you wish to use. The use of a complex denominator will not be covered here. We will only cover the three percents that are most commonly used:

PCTN - percent of total.

ROWPCTN – percent across row (use row total as denominator).

COLPCTN – percent across column (use column total as denominator).

This example shows the use of ROWPCTN. Note that the percents will add up to 100% (taking into account rounding errors) across each row. COLPCTN works similarly – the columns will add up to 100%.

```
PROC TABULATE data=one;
      CLASS ethnic educ;
      TABLE ethnic * ROWPCTN,
            Educ ;
RUN;
```

		Educ			
		High School	Bachelors	Masters	Doctorate
Ethnic					
Asian	RowPctN	25.00	75.00		
Af.Amer.	RowPctN	33.33	16.67	33.33	16.67
Hisp.	RowPctN	28.57	14.29	28.57	28.57
Am. Ind.	RowPctN		50.00	25.00	25.00
White	RowPctN	27.27	27.27	27.27	18.18

Special Things you can do with Formats – The MLF Option

Normally when you create a format, you can not have repeated values on the left side of the format definition. The Multilabel option in PROC FORMAT allows you to specify the same value to apply to different groups. To apply this in a PROC TABULATE, you must use the MLF (multi-label format) option on the CLASS statement for the variable you want multiple groups for. Note that the sort order will be alpha by the formatted values. If you want other orders, use the ORDER= option. Shown here is the ORDER=DATA option with a presort of the data by education level.

```
proc format;
      value $educmlf (multilabel)
            '1' = 'High School'
            '2','3','4'='College'
            '2' = 'Bachelors'
            '3' = 'Masters'
            '4' = 'Doctorate'
            ;
run;

proc sort data=one;
      by educ;
run;

proc tabulate data=one;
      class educ / mlf order=data;
      var income;
      tables educ='Education Level',
            Income='Income'*(N Mean);
      format educ $educmlf.;
run;
```

	Income	
	N	Mean
Education Level		
High School	8	45385.75
College	23	55663.83
Bachelors	9	44966.33
Masters	7	55670.14
Doctorate	7	69411.43

Special Things you can do with Formats – The PreLoadFMT Option

The PRELOADFMT option can be used in three ways depending on what other options you use with it. This option has no effect unless you use one of three other options with it AND you also assign formats to the class variable(s).

- Use with the PRINTMISS option to include all categories from the format in the table regardless of whether data exists for any individual category. This will allow you to get a 'complete' list of all categories in your output table whether or not you have data in that category. I find this one to be very, very useful. For example, you want to include all levels of ethnicity, but you might not have someone in each level. The table will create a row for that ethnicity, even if there isn't data. See example 1 below.

- Use with the EXCLUSIVE option to include ONLY those categories from the format in the table even if data exists for other values. This will cause some rows/columns to be removed from the table, even when data exists for those rows. See example 2 below.

- Use with ORDER=DATA if you want PROC TABULATE to use the order the levels in the table in the same order they appear in the PROC FORMAT. With this option, you will also need to use the (NOTSORTED) option in the PROC FORMAT VALUE statement. This allows you to create your own sort order to the data. See example 3 below.

Example 1: Use PRELOADFMT with the PRINTMISS option:

```
proc format;
      value $ethnic
            'W'='White'
            'H'='Hisp.'
            'I'='Am. Ind.'
            'A'='Asian'
            'B'='Af.Amer.'
            'L'='Alaskan Native'
            Other='Missing'
            ;
proc tabulate data=one;
      class ethnic / PRELOADFMT ;
      var income;
      tables ethnic,
                  Income*(N Mean)
                  / PRINTMISS ;
      format ethnic $ethnic.;
run;
```

	Income	
	N	Mean
Ethnic		
Asian	2	53474.00
Af.Amer.	5	58866.00
Hisp.	5	46563.40
Am. Ind.	4	62382.50
Alaskan Native	0	.
White	11	47694.27

Example 2: Use PRELOADFMT with the EXCLUSIVE option:

```
proc format;
      value $ethEXCL
            'H'='Hisp.'
            'I'='Am. Ind.'
            'L'='Alaskan Native'
            Other='Missing'
            ;
proc tabulate data=one;
      class ethnic / PRELOADFMT EXCLUSIVE;
      var income;
      tables ethnic,
                  Income*(N Mean);
      format ethnic $ethEXCL.;
run;
```

	Income	
	N	Mean
Ethnic		
Hisp.	5	46563.40
Am. Ind.	4	62382.50

Example 3: Use PRELOADFMT with the ORDER=DATA option:

```
proc format;
      value $ethnic (NOTSORTED)
            'W'='White'
            'H'='Hisp.'
            'I'='Am. Ind.'
            'A'='Asian'
            'B'='Af.Amer.'
            'L'='Alaskan Native'
            Other='Missing'
            ;
proc tabulate data=one ORDER=DATA;
      class ethnic / PRELOADFMT;
      var income;
      tables ethnic,
                  Income*(N Mean);
      format ethnic $ethnic.;
run;
```

	Income	
	N	Mean
Ethnic		
White	11	47694.27
Hisp.	5	46563.40
Am. Ind.	4	62382.50
Asian	2	53474.00
Af.Amer.	5	58866.00

Adding Formats to Stats and Removing Horizontal Lines

Let's take a look at two simple options to improve the table's look. The first option allows you to specify formats for the numbers in the cells of the table using the *F=fmt. expression. The second option is the NOSEPS option which removes the horizontal dividers between the row values from your table.

```
PROC TABULATE data=one NOSEPS;
      CLASS educ gender fulltime / missing;
      VAR income;
      TABLE educ=' ' * (fulltime=' ' ALL),
            gender=' ' * Income=' ' * ( N*F=6.  MEAN*F=Dollar8. );
RUN;
```

		Female		Male	
		N	Mean	N	Mean
High School	Fulltime			3	$52,507
	Parttime	3	$44,789	2	$35,600
	All	3	$44,789	5	$45,744
Bachelors	Fulltime	5	$42,771	0	
	Parttime	3	$36,947	0	
	All	8	$40,587	0	
Masters	Fulltime			4	$50,729
	Parttime	0		3	$62,258
	All	0		7	$55,670
Doctorate	Fulltime	5	$74,590	2	$56,466
	All	5	$74,590	2	$56,466

Clean up Row Headers with INDENT and RTS OPTIONS

Two more options to improve the look of your table are INDENT= which is very nice for subsetting row subheaders and RTS= which specifies how wide you want the row header field to be. Note, the RTS= count includes the | bar dividers at the beginning and the end of the row header fields, so include these in your count.

```
PROC TABULATE data=one NOSEPS;
      CLASS educ gender fulltime / missing;
      VAR income;
      TABLE educ=' ' * (fulltime=' ' ALL),
            gender=' ' * Income=' ' * ( N*F=6.  MEAN*F=Dollar8. )
            / BOX='Income' INDENT=3  RTS=12;
RUN;
```

Income	Female		Male	
	N	Mean	N	Mean
High School				
Fulltime			3	$52,507
Parttime	3	$44,789	2	$35,600
All	3	$44,789	5	$45,744
Bachelors				
Fulltime	5	$42,771	0	
Parttime	3	$36,947	0	
All	8	$40,587	0	
Masters				
Fulltime			4	$50,729
Parttime	0		3	$62,258
All	0		7	$55,670
Doctorate				
Fulltime	5	$74,590	2	$56,466
All	5	$74,590	2	$56,466

ODS Style Elements

ODS Style elements can also be used to change the look of a table. A few are listed below. These elements all work with HTML, PDF, RTF, PS and PCL destinations. Other elements are listed in the "SAS 9.1 Output Delivery System: User's Guide" that you can find at http://support.sas.com/documentation/onlinedoc/91pdf/ .

Foreground/Background=	modify color
BorderWidth=	specify thickness of borders
Just/Vjust=	specify horizontal and vertical justification
Font_Face/Font_Weight/Font_Size=	change font characteristics
Rules=	specify horizontal and vertical rule dividers
CellWidth/CellHeight=	change size of table cells
Cellpadding/CellSpacing=	specify white space and thickness of spacing around cell
OutputWidth=	specify width of table

Where do you put the Style elements?

Depending on where you place the style options, many different results can be achieved. If you place the style options on the PROC TABULATE statement, for example, you will affect all the table cells. See below for a list of some of the different places where you can put the style options and what portion of the table they will affect.
Note that for the CLASS, CLASSLEV, VAR, and KEYWORD statements, the style options can also be specified in the dimension expression in the Table statement.

Style Place In	**Part of Table Affected**
PROC TABULATE S=[…]	data cells
CLASS varname / S=[…]	heading for variable varname
CLASSLEV varname / S=[…]	class values for variable varname
VAR varname / S=[…]	heading for variable varname
KEYWORD stat / S=[…]	heading for named stat
TABLE page,row,col / S=[…]	table borders, rules, cell spacing
BOX={label=' ' S=[…] }	table Box

Changing the Foreground (Text) or Background Color

Here is an example showing how to change the color in various parts of the table. Notice that we are applying a black foreground (text) to all cells in the table. For the gender variable we are assigning a background of yellow to the values only (not to the column header) using the CLASSLEV statement. For the mean statistic label we are using a foreground of white and a background of purple.

```
ODS RTF file='c:\myfile.rtf';
PROC TABULATE data=one f=10.2 S=[foreground=black];
        CLASS gender;
        CLASSLEV gender / S=[background=yellow];
        VAR income;
        TABLE gender=' ' all={label='Tot'},
                        mean={s=[foreground=white background=purple]} * income
                /       box={label='Income'};
Run;
ODS RTF close;
```

Income	Mean
	Income
Female	52000.69
Male	54089.53
Tot	53011.42

Add Justification

This example shows how to add justification to three sections of the table. We are going to center all the cells inside the table, right justify the total 'Tot' row label, and Bottom Right justify the text inside the upper left box.

You can use ODS Style options to add many things to the table. Here is an example that adds color and justification. We are setting the foreground (print type) to be black and all the cells should be centered. Note the use of the new CLASSLEV statement that we have not seen before.

```
ODS RTF file='c:\myfile.rtf';
PROC TABULATE data=one f=10.2 S=[foreground=black just=c];
        CLASS gender;
        CLASSLEV gender / S=[background=yellow];
        VAR income;
        TABLE gender=' ' all={label='Tot' s=[just=R]},
                        mean={s=[foreground=white background=purple]} * income
                /       box={label='Income' s=[VJust=B Just=R]};
Run;
ODS RTF close;
```

	Mean
Income	Income
Female	52000.69
Male	54089.53
Tot	53011.42

Changing Cell Width

This example shows how to change the cell width of both the row headers and the columns of the table.

```
ODS RTF file='c:\myfile.rtf';
PROC TABULATE data=one f=10.2 S=[foreground=black just=c cellwidth=200];
        CLASS gender / S=[cellwidth=250];
        CLASSLEV gender / S=[background=yellow];
        VAR income;
        TABLE gender=' ' all={label='Tot' s=[just=R]},
                        mean={s=[foreground=white background=purple]} * income
                /       box={label='Income' s=[VJust=B Just=R]};
Run;
ODS RTF close;
```

	Mean
Income	Income
Female	52000.69
Male	54089.53
Tot	53011.42

Removing the Lines in the Table and Adjusting Cell Spacing

With the addition of the rules, cellspacing and cellpadding options we can remove all lines from the table.

```
ODS RTF file='c:\myfile.rtf';
PROC TABULATE data=one f=10.2 S=[foreground=black cellwidth=200 just=c];
        CLASS gender;
        CLASSLEV gender / S=[background=yellow];
        VAR income;
        TABLE gender=' ' all={label='Tot' s=[just=R]},
                        mean={s=[foreground=white background=purple]} * income
                /       s=[rules=none cellspacing=0 cellpadding=10]
                        box={label='Income' s=[VJust=B Just=R]};
Run;
ODS RTF close;
```

	Mean
Income	Income
Female	52000.69
Male	54089.53
Tot	53011.42

Adding a Flyover Text Box using ODS and Proc Format

You can use formats with styles to do a number of tricks. One thing you can do is to add a flyover text box to your row or column header cells. The result can not be demonstrated here, but what happens is when you hold your curser over the column header when you view the table in a Web Browser, a little text box will open underneath that shows the text you have in your format.

```
PROC FORMAT;
        VALUE $ethnic
                'W'='White - Non-Hispanic'
                'H'='Hispanic American'
                'I'='American Indian'
                'A'='Asian'
                'B'='African American'
                Other='Missing'
                ;
ODS HTML file='c:\myfile.HTML';
PROC TABULATE data=one;
        CLASS ethnic gender;
        CLASSLEV ethnic / s=[flyover=$ethnic.];
        VAR income;
        TABLE gender,
                ethnic * income;
RUN;
ODS HTML CLOSE;
```

Using Proc Format and ODS to Add Pictures to your Table

Another trick is to add a figure to the row or column header cell. Note that in this example, the gif files needed to be in the same directory that the HTML file was in.

```
PROC FORMAT;
        VALUE $gendGif
                'M'='bluebutterfly.gif'
                'F'='pinkbutterfly.gif';
RUN;

ODS HTML file='c:\myfile.HTML';
PROC TABULATE data=one;
        CLASS GENDER;
        CLASSLEV gender / S=[VJust=T postimage=$gendGif.  Cellwidth=80];
        VAR income;
        TABLE  income * (N Mean)
                INCOME * MEAN * GENDER;
RUN;
ODS HTML CLOSE;
```

Income		Income	
		Mean	
		Gender	
N	Mean	Female	Male
30	52111.80	52000.69	52238.79

Adding Colors to Cells Based on Values in the Cells

The next trick I am going to mention is how to use a format to highlight different cells with different colors based on the final value in the cell.

```
Title 'Highlighting Cell Values with Colors';
ODS HTML FILE='c:\myfile.html';
PROC FORMAT;
        VALUE watchit
                0 - 20000 = 'Green'
                20001 - 40000 = 'Orange'
                40001 - 50000 = 'Blue'
                50001 - 60000 = 'Purple'
                60001 - high = 'Red' ;
RUN;
PROC TABULATE data=one S=[foreground=watchit.] ;
        CLASS gender ethnic / MISSING;
        VAR income;
        TABLE gender ALL,
                Income * (ethnic ALL) * Mean;
RUN;
ODS HTML CLOSE;
```

	Income						
	Ethnic						All
	Missing	Asian	Af.Amer.	Hisp.	Am. Ind.	White	
	Mean	Mean	Mean	Mean	Mean	Mean	Mean
Gender							
Female	60398.50	26948.00	62083.67	49304.25	56945.00	43321.00	52000.69
Male	57147.50		54039.50	35600.00	78695.00	49334.25	52238.79
All	58773.00	26948.00	58866.00	46563.40	62382.50	47694.27	52111.80

Adding Colors to Row/Column headers Based on their Values

The last trick is how to change the background color of the row headers based on the value of the class variable.

```
Title 'Highlighting Row Headers with Colors';
ODS HTML FILE='c:\myfile.html';

PROC FORMAT;
        VALUE $gendbac
                'M'='blue'
                'F'='pink';
        VALUE $gendfor
                'M'='white'
                'F'='black';
RUN;

PROC TABULATE data=one;
        CLASS gender;
        CLASSLEV gender / S=[background=$gendbac. Foreground=$gendfor. Cellwidth=100];
        VAR income;
        TABLE gender ALL,
                Income *  (N Mean Std);
RUN;

ODS HTML CLOSE;
```

	Income		
	N	Mean	Std
Gender			
Female	16	52000.69	23861.47
Male	15	54089.53	20034.90
All	31	53011.42	21751.54

Conclusion and Bibliography

Proc Tabulate can greatly increase your ability to produce informative, beautiful, and professional looking tables. This paper covered many of the ways to produce a great table, but the options described here are by no means exhaustive of what you can do with PROC TABULATE.

There are several very good resources that you can use to learn more about PROC TABULATE. Among them are:

PROC TABULATE by Example, by Lauren Haworth
SAS Guide to Report Writing: Examples, by Michele M. Burlew
SAS Guide to TABULATE Processing, 1987 Edition, SAS Institute, Inc.

Author Contact

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

Wendi L. Wright
1351 Fishing Creek Valley Rd.
Harrisburg, PA 17112
Phone: (717) 513-0027
E-mail: wrightwendi6@gmail.com