

SESUG Paper RIV33-2017
**Using PROC REPORT and ODS STYLE Options
to Make Really Great Tables**

Wendi Wright, Questar Assessment, Inc.

ABSTRACT

Questar Assessment, Inc. had a need to automate many tables in a large public report created every year. This paper describes the techniques used to automate the tables in SAS®. The tables can now be created with little or no modification before being posted, making the creation of the report much more efficient.

This intermediate-level paper shows examples of how to use the ODS RTF STYLE= option in PROC REPORT to customize the borders in table headers and table cells. It will also cover the many other ways the STYLE option can be used to set widths, heights, and justification and to change fonts and colors in the table's foreground and background. Also covered will be how to add subscript, superscript, and carriage returns to cell contents.

INTRODUCTION

SAS, starting with version 9.2, has added the ability to customize table header borders and individual cell borders. One technique uses ODS RTF and the new Style= options with PROC REPORT. Two SAS Help Documents which show how to do this are available here:

<http://support.sas.com/kb/46/021.html>

<http://support.sas.com/kb/46/022.html>

This presentation will break down these two examples into steps and will present additional specific techniques and tricks. This presentation is not intended to teach the basics of PROC REPORT. Art Carpenter has an excellent paper on the Basics of PROC REPORT that teaches the basics:

PROC REPORT Basics: Getting Started with the Primary Statements
by Art Carpenter, Presented at SGF

<http://support.sas.com/resources/papers/proceedings12/242-2012.pdf>

BRIEF REVIEW OF BASIC SYNTAX

Sample basic code to use when using PROC REPORT appears below:

```
PROC REPORT <options>;  
  COLUMN column-specs / <options>;  
  DEFINE report-item / <options>;  
  COMPUTE report-item </ type_specification>;  
    CALL DEFINE (column_id (or _Row_),  
                 < 'attribute_name', value>);  
  ENDCOMP;  
  ... other statements ...  
RUN;
```

Syntax Explanation:

- The PROC REPORT statement has multiple options, one of which is the NOWD option. Note that this is now the default and is required in order for the new Style= options to work.
- The Column statement defines the columns you want to include in the report. These can be variables in the dataset, or variables computed inside the report procedure.
- The Define statement is used with each variable to define column formatting, special style options, labels, etc.
- The Compute statement is used to set up conditional formatting based on variables in the report and/or to compute new report items. These can be related to the column or can be applied to the row. The Endcomp statement ends the compute block. Note that new variables are not saved to the incoming dataset.
- Other statements include the BY statement to get reports by groups, the Break and Rbreak statements to add dividers between groups, and the Freq and Weight statements.

ODS CUSTOM STYLE TEMPLATE

A custom ODS RTF template was used in the examples in this paper. The purpose was to create formatting specific to work-specific requirements and to set up some base values to use in all of the tables. The margins and fonts were modified and a special option called PROTECTSPECIALCHARS was set to OFF. This is a required change in order for the new Style Options to work correctly. When the special characters option is set to OFF, it allows the use of a special character to pass style commands to the RTF destination. We will use the ~ symbol to call in some of the style options from the COLUMN statement. To define the ~ symbol we use the command ODS EscapeChar='~'. The code used to set up the Template is below:

```
ODS path(prepend) maclib.templat(update);

PROC TEMPLATE;
  Define style MyRTF;
    Parent=styles.rtf;
    Replace Body from Document /
      bottommargin=1in
      topmargin=1in
      rightmargin=1in
      leftmargin=1in ;
  Replace Fonts /
    'TITLEFont2' = ("Times New Roman",11pt,Bold)
    'TITLEFont' = ("Times New Roman",11pt,Bold)
    'StrongFont' = ("Times New Roman",11pt,Bold)
    'EmphasisFont' = ("Times New Roman",11pt,Italic)
    'FixedEmphasisFont' = ("Courier",11pt,Italic)
    'FixedStrongFont' = ("Courier",11pt,Bold)
    'FixedHeadingFont' = ("Courier",11pt,Bold)
    'BatchFixedFont' = ("SAS Monospace, Courier",6.7pt)
    'FixedFont' = ("Courier",11pt)
    'headingEmphasisFont' = ("Times New Roman",10pt,Bold)
    'headingFont' = ("Times New Roman",10pt,Bold)
    'docFont' = ("Times New Roman",10pt) ;
  Replace Document /
```

```

PROTECTSPECIALCHARS = OFF
linkcolor = colors('link2')
activelinkcolor = colors('link1')
visitedlinkcolor = colors('link1');
End;      * of Define Style MyRTF;
Run;

```

ODS ESCAPECHAR

When the PROTECTSPECIALCHARS option is set to OFF, it allows the use of a special character to pass style commands to the RTF destination. To use the ODS Escapechar options, first you have to specify what your escape character is going to be. Here we are using tilde '~', but you can choose any character you want. Other common escape characters are the backslash, forward slash, and caret '^'. Here we are using the tilde because it is seldom seen in English text. As we move further into our examples, we will demonstrate how to have total control over all of the cell dividers in the column headers using the ODS EscapeChar. Example code is below:

```
ODS ESCAPECHAR='~';
```

In addition to the ODS EscapeChar style options, the use of the special escapechar allows the use of inline style attributes. These add the ability to do underlining, superscript, subscript, bolding, italics, etc. Some of these options will be shown in a later example.

EXAMPLE 1– A LOOK AT NO STYLE OPTIONS VS WITH STYLE OPTIONS

NO STYLE OPTIONS

Our first example shows a table with no style options. The PROC REPORT code below produces a table with three columns that shows how many students were dropped during data cleaning. The code for Example 1 is shown below:

```

PROC REPORT data=report nowd;
  COLUMN exclusion del left;
  DEFINE exclusion / display 'Exclusion Rule';
  DEFINE del      / display  '# Deleted';
  DEFINE left     / display  '# Cases Remain';
  TITLE Example 1. Data Cleaning Results;
RUN;

```

Example 1. Data Cleaning Results

Exclusion Rule	# Deleted	# Cases Remain
Initial Number of Cases	n/a	200
Missing ID	15	200
Invalid Code	0	185
Duplicated Record	2	183
Out-of-Range	12	171
Incorrect Subject	6	165

WITH STYLE OPTIONS ADDED

To the table above, we used Style options to make the following modifications. The code will be shown later in the paper.

- removed left and right borders of the table.
- adjusted the width of the columns to make them wider.
- right justified the first column and centered the others.
- shaded the header row a lighter gray using a custom color.
- adjusted the padding between the first column and the column divider on the right.

Example 1. Data Cleaning Results

Exclusion Rule	# Deleted	# Cases Remain
Initial Number of Cases	n/a	200
Missing ID	15	185
Invalid Code	0	185
Duplicated Record	2	183
Out-of-Range	12	171
Incorrect Subject	6	165

FIRST EXAMPLE – EXPLANATION OF STYLE OPTIONS USED**STYLE OPTIONS FOR THE FULL TABLE**

There are two ways to add style options to your report - the Style= option and the Style Escapechar. These are called Style Modifiers. Many style modifier options are only for certain destinations. This paper will focus on the modifiers for the RTF destination. At last count, there are over 100 different style modifier options. Depending on where you put your Style modifiers, they will affect various parts of the

report. See Appendix A for the table from the SAS Help Documents that shows what is affected depending on placement of the options.

The Style Modifiers in the PROC REPORT statement may be assigned at two levels:

- Options in the STYLE(REPORT)= section will affect the entire table; and
- Options in the STYLE(HEADER)= section will affect the header rows of the table.

The code we used is below:

```
PROC REPORT data=report nowd
  STYLE(report) = [rules=none frame=hsides
    cellspacing=0 cellpadding=0]
  STYLE(header) = [foreground=black background=cxD9D9D9
    cellheight=.23in
    borderbottomwidth=.5pt borderbottomcolor=black
    bordertopwidth=.5pt bordertopcolor=black
    borderleftwidth=.5pt borderleftcolor=cxD9D9D9
    borderrightwidth=.5pt borderrightcolor=cxD9D9D9]
;
```

Style (Report)

The STYLE(report) option will control the default options for your entire table. The Rules= option controls default lines inside the table, and Frame= controls what lines will be around the outside of the table. We chose to use Frame=Hsides to put lines above and below the table but not on the left and right sides. Since we will be specifically telling SAS where to put lines inside the table, we are choosing to turn off the automatic placement of the lines by selecting RULES=None.

- Rules= where you want lines inside your table. Choices are: All, Cols, Rows, None, Groups.
- Frame= where you want lines around your table. Choices are: Above, Below, Box, Hsides, Vsides, Void (no borders), LHS (left side only), RHS (right side only).

The Cellpadding= and Cellspacing= options are used to specify how much spacing is around the cell borders. As a default we are choosing to put 0 for both and will later override that in a few places where we need to.

- Cellpadding – amount of white space on each of the four sides of the content in a cell.
- Cellspacing (or Borderspacing) – specifies vertical and horizontal thickness of the spacing between cells.

STYLE OPTIONS FOR THE HEADER

There are two ways to specify styles for your table header. One is to set up defaults on the PROC REPORT statement and the other is to custom specify each column header using the COLUMN statement.

Using PROC REPORT to specify Header Styles

The STYLE(header) options specified on the PROC REPORT statement control the default options for the header part of the table.

- Foreground (or Color) – the color of the text.
- Background - the color of the shading in the background (the shading we are using for these

- tables is cxD9D9D9 (RGB color specification).
- Cellheight (or Height) – specifies the height of the rows in the header.
- LineStyle –specifies solid or dashed lines.
- LineThickness – how thick you want the line.

Border options allow you to specify, for each cell, the width, color, and style for the top, bottom, left, and right borders. My company uses a borderwidth of .5pt and a shading color of cxD9D9D9 for the headers. For the borders, black is used to show borders and cxD9D9D9 to match the shading and make the border invisible. For the table cells (we will discuss this more later), the colors of white or black are used (white will make the cell borders invisible).

To set options for all four sides at once, you can use BorderWidth=, BorderColor=, and BorderStyle=. BorderWidth is not in the SAS Help documentation; however, it works without errors when tested.

Border options allow us to specify width, color, and style for each border (Bottom, Top, Right, and Left). Below we show 'TOP':

- BorderTOPWidth=.
- BorderTOPColor=.
- BorderTOPStyle = (linestyle=).

Border Width can be specified in:

- cm – centimeters.
- em – standard typesetting unit for width.
- ex – standard typesetting unit for height.
- in – inches.
- mm – milliliters.
- pt – printer's point.

Border Color can be defined in any of the following ways:

- a predefined SAS color (for example, blue or VIYG).
- a red/green/blue (RGB) value (for example, CX0023FF).
- a hue/light/saturation (HLS) value (for example, H14E162D).
- a gray-scale value (for example, GRAYBB).
- a red/green/blue transparency (RGBA) value (for example, a98FB9880) - not supported by Java devices.
- a cyan/magenta/yellow/black (CMYK) value (for example, FFFFFFF00).

There are many different font options that you can specify. We chose to specify the font colors in our custom template rather than in the PROC REPORT options, but note that you can use any of these in your program to override what is in the template. FontStyle and FontWeight are particularly useful if you want to emphasize something in your TITLE or header text. See Appendix B for the full list of available values.

- Font – specifies a font definition.
- TITLEFont=("<sans-serif>,<MTsans-serif>,Helvetica,Helv",3,bold).
- FontFamily (or font_face) - "Courier".
- FontSize - dimension or size.
- FontStyle - Italic/Roman/Slant.
- FontWeight - Bold/Medium/ others.
- FontWidth - Normal/Wide/ others.

Margins can also be specified in the options in PROC REPORT although in this paper, they are specified in the template. Specifying margins in the procedure will override those specified in the template. Style options to set margins for this particular table are:

- MarginTop
- MarginBottom
- MarginLeft
- MarginRight

These are all useful options to specify what you want as defaults for the headers in your table. If you want to override these defaults and specify other options for only one or a few columns, use the Style EscapeChar modifiers on the COLUMN statement.

Using COLUMN Statement to Specify Styles

The Column statement does not have a STYLE= option that is available for use. Instead, to specify customized style options for each of the column headers in the table, use the Style EscapeChar modifiers. This allows A LOT of customization in the headers of the table. You can specify the text you want in the headers here, the background color, any dividers or rules, and what grouped headers should look like, as well as many other options.

SYNTAX IS EXTREMELY IMPORTANT!

- Parentheses - define columns in the header – a 'column' is defined as where there is a divider from the topmost column to the first row of the table. This will be clearer when we get to multi-row tables.
- Quotes – around the entire Style specification.
- Brackets – around the Style options.
- After the closing bracket is the header label you want to use in the table and following this is the closing quote.
- After the closing quote is/are the variable(s) from your input dataset that feed into that column. Note that if two variables are listed here then the table will contain two columns below that header, three variables listed equals three columns, etc.

The code for the COLUMN statement is below. Highlighted is the header label and header variable for each of the three columns:

```
COLUMN
(
  ('~S={borderbottomcolor=black borderbottomwidth=.5pt
    borderrightcolor =black borderrightwidth =.5pt
    borderleftcolor  =cxD9D9D9 borderleftwidth=.5pt
    cellheight=.23in }Exclusion Rule' exclusion )
  ('~S={borderbottomcolor=black borderbottomwidth=.5pt
    borderrightcolor =black borderrightwidth =.5pt
    cellheight=.23in }# Deleted' del )
  ('~S={borderbottomcolor=black borderbottomwidth=.5pt
    borderrightcolor =cxD9D9D9 borderrightwidth =.5pt
    cellheight=.23in }# Cases Remain' left )
);
```

Note the parentheses that define the three columns. One set of parentheses is around the definition for that column's header. Also used is an additional set of parentheses around all three column style definitions. The author found that this was required in some cases in order for the styles to work, but not in every case or for every table. Also note how the quotes and brackets are set up and see how the header label is separated from the variable name.

Below is the table again that we have been looking at. These options on the Column statement will only affect the header portion of the table. Notice the options for the borders. We have borders between the columns in the header, but not at the two ends. So, you can see that the first column is specified to use Borderleftcolor of the same shade as the background color. Similarly, the last column is specified to use Borderrightcolor= the shading color.

This will get more complicated when there are headers with two or three rows and the columns are subdivided multiple times.

Example 1. Data Cleaning Results

Exclusion Rule	# Deleted	# Cases Remain
Initial Number of Cases	n/a	200
Missing ID	15	185
Invalid Code	0	185
Duplicated Record	2	183
Out-of-Range	12	171
Incorrect Subject	6	165

STYLE OPTIONS FOR THE TABLE CELLS

There are two ways to specify styles for your table header. One is to set up defaults on the PROC REPORT statement (which we have already discussed) and the other is to custom specify each column header using the DEFINE statement.

Using the DEFINE Statement to Specify Styles

The Define statement is used to control the formatting for the data cells inside the table. The Define statement DOES have a STYLE= option so that the option syntax will look more like what we saw in the PROC REPORT statement. We are using the Style location of Column here, which will affect the Column cells inside the table itself (the data cells). We can control the borders (all four sides of each cell) as well as justification (both horizontal and vertical), the padding values, etc. Later an example will be shown for how to conditionally set options for rows using a COMPUTE block. Example code is below:

```
DEFINE exclusion / display ' '
  style(column) = {cellheight=.23in cellwidth=2.0in
    vjust=middle just=right
    borderleftcolor=white borderleftwidth=.5pt
    borderrightcolor=black borderrightwidth=.5pt paddingright=.1in};
```

```

DEFINE del / display center ' '
               style(column)={cellheight=.23in cellwidth=0.7in vjust=middle
                               borderleftcolor =black borderleftwidth =.5pt
                               borderrightcolor=black borderrightwidth=.5pt};

DEFINE left / display center f=comma10. ' '
               style(column)={cellheight=.23in cellwidth=1.5in vjust=middle
                               borderleftcolor =black borderleftwidth =.5pt
                               borderrightcolor=white borderrightwidth=.5pt};

```

Center, VJust and Just (yellow highlight) will set the justification for the columns. Options for Just are Center, Dec, Left, and Right. Options for VJust are Bottom, Middle, and Top.

Border color and width options are the same as we've seen before. However, note that the BorderLeftColor=White for the leftmost column and BorderRightColor=White for the rightmost column. This will in effect remove the borders on the left and right sides of the cells in the table. Note that this does NOT affect the column headers.

Cellwidth determines the width of the column. This can be specified in several different units; Inches 'in' is used above.

Padding values add space between the text and the cell divider. Remember the default for cellpadding and cellspacing was 0. Now we want to override that for the first column. If we have some text right or left justified and cellpadding is 0, the text will be jammed up right next to the column divider. To make it look nicer and have a little space between the two, we can use the paddingright= or paddingleft= options. Often left justification is combined with paddingleft and right justification is combined with paddingright. PaddingTop and PaddingBottom are used depending on if VJUST= top or bottom, respectively.

Below is a copy of the table we have been using. Note how the data rows look based on our options above.

Example 1. Data Cleaning Results

Exclusion Rule	# Deleted	# Cases Remain
Initial Number of Cases	n/a	200
Missing ID	15	185
Invalid Code	0	185
Duplicated Record	2	183
Out-of-Range	12	171
Incorrect Subject	6	165

EXAMPLE 2

The next example features a two row header with conditional row dividers inside the table and we again see the use of PaddingRight= and PaddingLeft=. The conditional column dividers will be produced by using a compute block.

Demographic Category		N-Count	Cronbachs Alpha		Feldt-Raju Coefficient	
			Est.	SEM	Est.	SEM
State	All Items	1000	0.55	2.01	0.44	2.20
Gender	Female	500	0.33	2.02	0.33	2.19
	Male	500	0.44	2.03	0.55	2.18
Ethnicity	Asian	100	0.44	2.04	0.66	2.17
	Black	100	0.55	2.05	0.77	2.16
	Hispanic	50	0.44	2.06	0.55	2.15
	American Indian	25	0.33	2.07	0.55	2.14
	Multiracial	25	0.22	2.08	0.66	2.13
	Pacific Islander	50	0.22	2.09	0.33	2.12
	White	650	0.55	2.10	0.44	2.11

EXAMPLE 2 – COLUMN STATEMENT OPTIONS (DEFINING THE HEADERS AT THE TOP)

When we get into multiple row headers, there is a special way that SAS looks at the columns of the header. When defining the Styles for the header, we need to define the header by Column, NOT by Row. For example, this table has two header rows and seven data columns beneath the header rows. However, when we set up the styles, we will set up four Header Column definitions. SAS recognizes the header by column only and defines all rows within each Header Column separately. This table has four 'Header' columns. The first column has two variables under the one header, one column with only one data variable below it, and the last two columns are grouped headers that each are further sub-divided into two columns. Each of these four columns will have its own STYLE option on the COLUMN statement.



Demographic Category		N-Count	Cronbachs Alpha		Feldt-Raju Coefficient	
			Est.	SEM	Est.	SEM
State	All Items	1000	0.55	2.01	0.44	2.20

First Column

The first column is one header cell but there are two data cells below it. However, we only want one label over both of the data columns. The SAS code for the start of the COLUMN statement and the first column is below:

```

COLUMN
(
  ('~S={borderbottomcolor=cxD9D9D9 borderbottomwidth =.5pt
    borderleftcolor =cxD9D9D9 borderleftwidth =.5pt
    borderrightcolor =black borderrightwidth =.5pt
    cellheight=.23in } '
    ('~S={borderbottomcolor =black borderbottomwidth =.5pt
      borderleftcolor =cxD9D9D9 borderleftwidth =.5pt
      bordertopcolor =cxD9D9D9 bordertopwidth =.5pt

```

```

                                borderrightcolor =black      borderrightwidth  =.5pt
                                cellheight=.23in } 'Demographic Category' ByVar ByValue
                                )
                                )

```

Note the beginning parentheses (first arrow) for the full set of column definitions. Then there is a parentheses set for the overall column and the second row inside that has its own set of parentheses. Syntax is very important when defining styles.

There is a separate '~S' style option for each row of the header. Note that between the two rows of the header we are making that border the same color of the general background (BorderBottomColor in the first row definition and BorderTopColor in the second row definition).

There is no label that goes on the first row of the header, so see that the end of the first row definition does not include any text label. The label for this column goes on the second row ('Demographic Category') and the two variables that will be placed as columns below this label are listed at the end of the second definition (ByVar and ByValue).

Then we close the two row definitions using the two closing parentheses. However, we do NOT close the overall table definition parentheses yet.

Second Column

The second column has only one row of data below it, but has two rows to define. The inside border between the two rows will be shaded cxD9D9D9 (invisible) and the outside borders of the cell are all black. The label for the column is on the second row and is 'N-Count' and the data variable is ncount. If you wanted to put the label on the first row, you could easily do so; just add or move text to the first row inside the quote at the end of the definition. The code for the second column is below:

```

('~S={ borderbottomcolor=cxD9D9D9 borderbottomwidth =.5pt
        borderleftcolor  =black      borderleftwidth  =.5pt
        borderrightcolor =black      borderrightwidth =.5pt
        cellheight=.23in } '
        ('~S={borderbottomcolor=black      borderbottomwidth=.5pt
            borderleftcolor = black      borderleftwidth  =.5pt
            bordertopcolor  =cxD9D9D9    bordertopwidth=.5pt
            borderrightcolor =black      borderrightwidth =.5pt
            cellheight=.23in }N-Count' ncount
        )
    )

```

Third Column

In the third column the first row is followed by a second row that is sub-divided into two separate columns, both with their own label. There is a label on the first row, and labels for both of the sub-set columns. Note that the data variables are specified on the second (or bottom) row of the definition, although there are labels on both rows.

There is one overall set of parentheses for the definition of the first row of that column, and then two sub-setting columns in the second row – each with their own set of parentheses. Each of the two subset

columns has its own label and its own data variable. Note that in the header, the right and left colors are set up so the divider between those two labels is invisible.

This technique can be used to define even more rows and even more sub-divisions. Just be careful of how you set up the parentheses. The code for the third column is below:

```
(~S={ borderbottomcolor =black   borderbottomwidth =.5pt
      ... Border style definitions ...
      cellheight=.23in } Cronbachs Alpha"
  (~S={borderrightcolor=cxD9D9D9   borderrightwidth=.5pt
      ... Other border style definitions ...
      cellheight=.23in } Est.' alpha
    )
  (~S={borderleftcolor=cxD9D9D9   borderleftwidth=.5pt
      ... Other border style definitions ...
      cellheight=.23in } SEM' sem
    )
  )
)
```

Fourth Column

The fourth column is set up similar to the third column, with one column definition for the top row, then two sub-setting column definitions in the second row. This is the last column for this table, so note that we end the COLUMN statement with a closing parentheses and the semicolon. The code for this is below:

```
(~S={ borderbottomcolor =black       borderbottomwidth =.5pt
      Borderrightcolor   =cxD9D9D9   borderrightwidth=.5pt
      .... Border style definitions ...
      cellheight=.23in } Feldt-Raju Coefficient "
  (~S={borderrightcolor=cxD9D9D9   borderbottomwidth=.5pt
      .... Border style definitions ...
      cellheight=.23in } Est.' feldt_raju
    )
  (~S={borderleftcolor=cxD9D9D9   borderbottomwidth=.5pt
      boderrightcolor=cxD9D9D9   borderrightwidth=.5pt
      .... Border style definitions ...
      cellheight=.23in } SEM' sem_feldt
    )
  )
);
```



Demographic Category		N-Count	Cronbachs Alpha		Feldt-Raju Coefficient	
			Est.	SEM	Est.	SEM
State	All Items	1000	0.55	2.01	0.44	2.20
Gender	Female	500	0.33	2.02	0.33	2.19
	Male	500	0.44	2.03	0.55	2.18
Ethnicity	Asian	100	0.44	2.04	0.66	2.17
	Black	100	0.55	2.05	0.77	2.16
	Hispanic	50	0.44	2.06	0.55	2.15
	American Indian	25	0.33	2.07	0.55	2.14
	Multiracial	25	0.22	2.08	0.66	2.13
	Pacific Islander	50	0.22	2.09	0.33	2.12
	White	650	0.55	2.10	0.44	2.11

EXAMPLE 2 – DEFINE STATEMENT OPTIONS (DEFINING THE DATA CELLS)

There is one new technique with this table that has not been covered yet. There is a conditional row divider after each demographic category. The other two techniques are familiar – the use of `PaddingLeft` and `PaddingRight` and using `BorderLeftColor` and `BorderRightColor` to make the border 'invisible'.

There are several things to note in the following code:

- The use of the `Just=` and the `PaddingLeft` and `PaddingRight` options in the `ByVar` and `ByValue` definitions.
- The formats for the numeric variables.
- The `BorderRightColor` and `BorderLeftColor` use that will show or not show the column.
- The different widths used for the table columns. The widths are customized to accommodate the length of the column header labels as well as the data lengths.

```
define byvar / display ' '
    style(column)={cellheight=.23in cellwidth=.8in
        vjust=middle just=left
        borderleftcolor=white borderleftwidth=.5pt
        borderrightcolor=black borderrightwidth=.5pt
        PADDINGLEFT=.1in};

define byvalue / display ' '
    style(column)={cellheight=.23in cellwidth=1.35in
        vjust=middle just=right
        borderleftcolor=black borderleftwidth=.5pt
        borderrightcolor=black borderrightwidth=.5pt
        PADDINGRIGHT=.1in};

define ncount / display ' ' f=comma8.
    style(column)={cellheight=.23in cellwidth=.67in
        vjust=middle just=center
        borderleftcolor=black borderleftwidth=.5pt
        borderrightcolor=black borderrightwidth=.5pt };

define alpha / display ' ' f=5.2
    style(column)={cellheight=.23in cellwidth=.55in
        vjust=middle just=center
        borderleftcolor=black borderleftwidth=.5pt
        borderrightcolor=white borderrightwidth=.5pt };
```

```

define sem      / display ' ' f=5.2
                  style(column)={cellheight=.23in cellwidth=.67in
                                vjust=middle just=center
                                borderleftcolor=white borderleftwidth=.5pt
                                borderrightcolor=black borderrightwidth=.5pt };

define feldt_raju / display ' ' f=5.2
                  style(column)={cellheight=.23in cellwidth=.80in
                                vjust=middle just=center
                                borderleftcolor=white borderleftwidth=.5pt
                                borderrightcolor=white borderrightwidth=.5pt };

define sem_feldt / display ' ' f=5.2
                  style(column)={cellheight=.23in cellwidth=.80in
                                vjust=middle just=center
                                borderleftcolor=white borderleftwidth=.5pt
                                borderrightcolor=white borderrightwidth=.5pt };

```

Example 2 – Getting Conditional Row Dividers

In order to get conditional row dividers, some prep-work had to be made to the incoming dataset, as well as to the PROC REPORT code.

We had to modify the incoming dataset to the PROC REPORT procedure. The modifications included adding a new variable to indicate the last row of each demographic section. In order to do this, the dataset was pre-sorted in print order, and the temporary variable 'LAST.byvar' was used to create a new variable called Rowline. Rowline is set to 1 when LAST.byvar is true.

Note the use of the NotSorted option on the By statement. This allows us to have our sort order in a different order than Byvar, but still make use of the Byvar groupings to indicate where a line should go. The SAS code is below:

```

PROC SORT data=reliabdemog;
    BY order order2 order3;
RUN;
DATA report;
    SET report;
    BY byvar NOTSORTED;
    IF LAST.byvar THEN rowline=1;
RUN;

```

Then we changed our Column statement in PROC REPORT. The name of the variable is placed somewhere in the column definition (outside of any of the parentheses). The SAS code is below:

```

COLUMN rowline
(
    ... rest of column statement ...

```

Next we created a DEFINE statement for the Rowline variable. Note the NoPrint option to prevent it from printing in the table and the ANALYSIS option which will tell the program to look for this variable in a Compute block. The SAS code is below:

```
DEFINE ROWLINE / ANALYSIS NOPRINT;
```

Now we added a Compute block. The Compute block contains conditional instructions to apply based on what is inside the compute block. This Compute block looks to see if the sum of the Rowline variable = 1 and if so, then use the CALL DEFINE statement to modify the style of the row. The style change we are using is to set BorderBottomColor=black. The ENDCOMP statement ends the compute block. The SAS code is below:

```
COMPUTE rowline;
  if rowline.sum=1 then
    CALL DEFINE (_row_,'style',
      'style=[borderbottomcolor=black
        borderbottomwidth=.5pt] ' );
ENDCOMP;
```

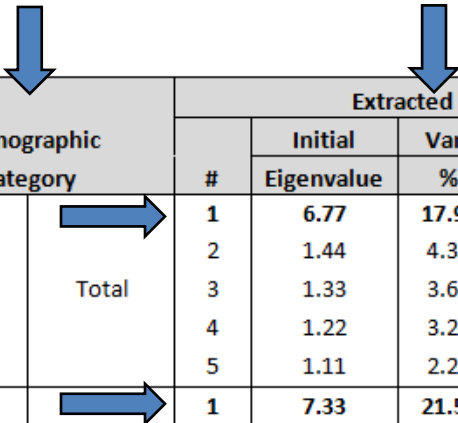
This is the completed table. There are a few things to note here.

- First column is justified left with a paddingleft value and, similarly, the second column is justified right with a paddingright value.
- No column dividers inside the data cells for the last two pairs of columns.
- Conditional row dividers between each demographic category.

Demographic Category		N-Count	Cronbachs Alpha		Feldt-Raju Coefficient	
			Est.	SEM	Est.	SEM
State	All Items	1000	0.55	2.01	0.44	2.20
Gender	Female	500	0.33	2.02	0.33	2.19
	Male	500	0.44	2.03	0.55	2.18
Ethnicity	Asian	100	0.44	2.04	0.66	2.17
	Black	100	0.55	2.05	0.77	2.16
	Hispanic	50	0.44	2.06	0.55	2.15
	American Indian	25	0.33	2.07	0.55	2.14
	Multiracial	25	0.22	2.08	0.66	2.13
	Pacific Islander	50	0.22	2.09	0.33	2.12
	White	650	0.55	2.10	0.44	2.11

EXAMPLE 3

Example 3 features a three row header (but only two columns to define), and conditional row dividers and conditionally bolded rows in the table. The two columns that we have to define are shown by the vertical arrows below. You can see that the second column is sub-divided many times. The horizontal arrows show the bolded rows inside the table.



Demographic Category		Extracted Factor			
		#	Initial	Variance Accounted for	
			Eigenvalue	%	Cumulative %
Total	Total	1	6.77	17.99	17.99
		2	1.44	4.33	21.00
		3	1.33	3.66	24.00
		4	1.22	3.22	27.00
		5	1.11	2.22	29.00
Gender	Female	1	7.33	21.55	21.55
		2	1.55	4.44	25.00
		3	1.22	3.33	28.00
		4	1.05	2.22	30.00
		5	1.00	2.33	32.00

The full code for this table is shown in Appendix C. This section will only point out a few particular techniques. First is the Column definition. Note how the parentheses are set up to designate the columns and the sub-divisions of the columns. There are many subsetting parentheses. The syntax is pretty complicated for this table. The SAS code is below:

```

COLUMN rowline number
( /* start of full table definitions */
  ('~S={ <style options here > }'
    ('~S={ <style options here > }Demographic'
      ('~S={ <style options here > }Category' var val
        )
      )
    )
  ) /* end of first column definition */

  /* start of 'second' column definition */
  ('~S={<style options here > }Extracted Factor'

    /* first sub column */
    (
      ('~S={<style options here > }'
        ('~S={<style options here > }#' number
          )
        )
      )
    ) /* end of first sub-column */

```

```

        /* Second sub-column */
        (
            ('~S={ <style options here > } Initial'
              ('~S={<style options here > } Eigenvalue' eigenvalue
              )
            )
        ) /* end of second sub-column */

        /* Third sub-column (which has two sub-sub-columns) */
        (
            ('~S={<style options here > }Variance Accounted for'
              ('~S={<style options here > } %' proportion
              )
              ('~S={<style options here > } Cumulative %' cumulative
              )
            )
        ) /* end of third sub-column definition */
    ) /* end of 'second' column - Extracted-Factor - definition */
) ; /* end of full table definition */

```

The next technique in this example shows how to highlight the first row in each demographic category. This is done very similarly to the way the conditional row dividers are set. In this example, we are using a variable that already exists in the dataset (number) to determine when to bold. We then have similar changes in the PROC REPORT code.

- Both variables should be in the Column statement. Note that the bolded row variable we are using is already being printed. We just have to change the option DISPLAY to ANALYSIS
- Have a Define statement for both variables (note that we want Number to be displayed, but do not want Rowline to be displayed).
- Include both variables in a COMPUTE block.

The SAS code is below:

```

COLUMN rowline
    (...start of column statement with other header definitions ...)

    ('~S={borderbottomcolor=black borderbottomwidth=.5pt
        borderleftcolor =black borderleftwidth =.5pt }Extracted Factor'
      ( ('~S={borderbottomcolor=cxD9D9D9 borderbottomwidth=.5pt
        borderrightcolor =black borderrightwidth =.5pt
        cellheight=.23in } '
        ('~S={borderbottomcolor=black borderbottomwidth=.5pt
        borderleftcolor =black borderleftwidth =.5pt
        bordertopcolor =cxD9D9D9 bordertopwidth=.5pt
        borderrightcolor =black borderrightwidth =.5pt
        cellheight=.23in }## number
        ... rest of column statement ... );

    DEFINE rowline / ANALYSIS NOPRINT;
    DEFINE number / ANALYSIS ' ' center
        BorderLeftColor=black ... other style options ...;

    COMPUTE rowline;
        if rowline.sum=1 then call define ( _row_,'style',

```

```

                                'style=[borderbottomcolor=black borderbottomwidth=.5pt]');
ENDCOMP;
COMPUTE number;
    if number.sum=1 then call define (_row_,'style',
                                'style=[fontweight=bold]');
ENDCOMP;

```

INLINE FORMATTING FUNCTIONS

Inline formatting is a way to add superscripting, subscripting, and other textual modifications to the text in your table. As before, you will need to define the ODS EscapeChar. Then there are specific functions that can be used to get what you want.

The following will display '10th'. The 'super' function gives you superscript to the text following it. The function and the text to be superscripted are in a set of brackets. Again, syntax is very important.

```

ODS ESCAPECHAR='~';

COLUMN
('~S={borderbottomcolor=black borderbottomwidth=.5pt
    ...other style definitions...
    cellheight=.23in }10~{super th} p10
)

```

Here are a series of examples from the SAS Help Documents that show several of these functions. Note that here they are using the '^' as the escape character. These show some examples of how to get the Greek letters Alpha and Sigma as well as how to change colors and set nested formatting. There are many functions available. The SAS code is below:

```

SAS Example from
http://support.sas.com/documentation/cdl/en/ODSug/69832/HTML/default/viewer.htm#p11xia2ltavr8ln17srq8vn4rngc.htm

ODS escapechar="^";  ODS RTF file="rtfInlinFuncs.rtf";
TITLE "Examples of Inline Formatting Functions";
TITLE2 'Example of ^{nbspspace 3} Non-Breaking Spaces Function';
TITLE3 'Example of ^{newline 2} Newline Function';
TITLE6 'Example ^{style [foreground=red] of Super,
    Alpha ^{super ^{unicode ALPHA}
    ^{style [foreground=green] Nested}} Formatting} and Scoping';
TITLE7 'Example of SUB, ^{sub ^{style [foreground=red] red
    ^{style [foreground=green] green } and
    ^{style [foreground=blue] blue styles }}}
    and SIGMA ^{unicode SIGMA} Functions';

proc print data=sashelp.class(obs=4); run;
ODS _all_ close;

```

The titles from running the code above are shown below.

Examples of Inline Formatting Functions
Example of Non-Breaking Spaces Function
Example of
Newline Function

Example of Super, Alpha ^α Nested Formatting and Scoping
Example of SUB, red green and blue styles and SIGMA σ Functions

RICH TEXT FORMAT (.RTF) CONTROL WORDS

EXAMPLE 1

One use of RTF Control Words is to add line breaks inside the cells of your table. This will require some pre-work in your dataset to set up the data variables correctly. Below is the first two rows of the table. Note how each cell has up to two line breaks in it.

Reporting Subscores and Total Subscore Points			
Grade	Subscore 1	Subscore 2	Subscore 3
10	Advanced Algebra 15	Number and Fractions 15	Geometry 15
11	Trigonometry 15	Proportional Relationships 15	Measurement And Data 15

In the code below, note the use of the '~n' which tells Word to put in a line break:

```
ODS EscapeChar='~';
Data ...;
...;
if subscore= 'AA' then sub1=
    'Advanced~n Algebra~n' || put(totpoints,2.);
else if subscore='NF' then sub2=
    'Number and~n Fractions~n' || put(totpoints,2.);
else if subscore= 'GE' then sub3=
    'Geometry~n' || put(totpoints,2.);
...
RUN;
```

EXAMPLE 2

Another example shows how to add highlighting to specific cells. This example shows how to highlight cells based on the value of another variable. Here we are using three shades of green or no highlight at all.

The syntax to use these with this is very specific.

- Use only a capital 'S' (a lower case 's' will not work).
- Use only the curly brackets around the style options.

In the code below, we are only showing the definition of one column (Cell1), but in the actual table, we set up multiple columns.

```
DATA ...;
  ...;
  if shade1=0 then cell1=strip(mytext);
  else if shade1=1 then
    cell1="~S={background=lightgreen" || strip(mytext);
  else if shade1=2 then
    cell1="~S={background=green
           foreground=white}" || strip(mytext);
  else if shade1=3 then
    cell1="~S={background=darkgreen
           foreground=white}" || strip(mytext);
  ...;
RUN;

PROC REPORT;
  Columns cell1 cell2 cell3;
  Define ...;
RUN;
```

In addition to the above examples, here is a link to the Microsoft area where they have a list of the many RTF control words.

<https://www.microsoft.com/en-us/download/details.aspx?id=10725>

A few of them are:

- ~ul for underline (\ul0 to turn off)
- ~b for bold (\b0 to turn off)
- ~i for italics (\i0 to turn off)
- ~sub for subscript
- ~super for superscript

CONCLUSION

The options described here are by no means exhaustive of what can be done with SAS Style Options. The options can be used with other procedures besides PROC REPORT and with other destinations besides RTF. There are many different Style options yet to discover. The author is only just starting to discover all of the uses of Inline Formatting and RTF Control Words.

REFERENCES

Arthur L. Carpenter, 2012. "PROC REPORT Basics: Getting Started with the Primary Statements"
<http://support.sas.com/resources/papers/proceedings12/242-2012.pdf>

SAS Help Documents, "PROC REPORT Statement"
<http://support.sas.com/documentation/cdl/en/proc/69850/HTML/default/viewer.htm#p0bqogcics9o4xn17yv12qjbgdpi.htm>

SAS Help Documents, "All Style Options"
<http://support.sas.com/documentation/cdl/en/ODSproc/69834/HTML/default/viewer.htm#n19a4b40swc766n18qczor47r08f.htm>

SAS Help Documents, "Sample 46021: Customize header borders in ODS PDF and ODS RTF output with PROC REPORT"
<http://support.sas.com/kb/46/021.html>

SAS Help Documents, "Sample 46022: Customize table or cell borders in ODS PDF and ODS RTF output using PROC REPORT"
<http://support.sas.com/kb/46/022.html>

SAS Help Documents, "ODS ESCAPECHAR Statement"
<http://support.sas.com/documentation/cdl/en/ODSug/69832/HTML/default/viewer.htm#p11xia2ltavr8ln17srq8vn4rnqc.htm>

CONTACT INFORMATION

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

Wendi Wright
1351 Fishing Creek Valley Rd
Harrisburg, PA 17112
(c) 717-421-9315
wwright@questarai.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.

APPENDIX A

STYLE<(location(s))>=<style-override(s)>

specifies one or more style overrides to use for different parts of the report.

location(s)

identifies the part of the report that the STYLE= option affects. The following table shows what parts of a report are affected by values of *location*.

The valid and default values for *location* vary by what statement the STYLE= option appears in. The following table shows valid and default *location* values for each statement. To specify more than one value of *location* in the same STYLE= option, separate each value with a space.

Locations and Default Style Elements for Each Statement in PROC REPORT

Statement	Valid Location Values	Default Location Value	Part of Report Affected	Default Style Element
PROC REPORT	COLUMN HEADER HDR SUMMARY REPORT LINES CALLDEF	REPORT	Report as a whole	Table
BREAK	SUMMARY LINES	SUMMARY	Summary lines	DataEmphasis
CALL DEFINE	CALLDEF	CALLDEF	Cells identified by a CALL DEFINE statement	Data
COMPUTE	LINES	LINES	Lines generated by LINE statements	LineContent
DEFINE	COLUMN HEADER HDR	COLUMN HEADER	Column cells Column headings	COLUMN: Data HEADER: Header
RBREAK	SUMMARY LINES	SUMMARY	Summary lines Lines generated by LINE statements	DataEmphasis

APPENDIX B

Font Keywords

Keywords for Font Weight	Keywords for Font Style	Keywords for Font Width
MEDIUM	ITALIC	NORMAL 1
BOLD	ROMAN	COMPRESSED 1
DEMI_BOLD 1	SLANT	EXTRA_COMPRESSED 1
EXTRA_BOLD 1		NARROW 1
LIGHT		WIDE 1
DEMI_LIGHT 1		EXPANDED 1
EXTRA_LIGHT 1		
1 Few fonts honor these values.		

APPENDIX C

```

PROC REPORT nowd data=report_subgroup
  style(report)=[rules=none frame=hsides cellspacing=0 cellpadding=0]
  style(header)=[foreground=black background=cxD9D9D9 cellheight=.23in
    borderbottomwidth=.5pt borderbottomcolor=black
    bordertopwidth=.5pt bordertopcolor=black];

COLUMNS rowline
(
  ('~S={borderbottomcolor=cxD9D9D9 borderbottomwidth=.5pt
    borderrightcolor =black borderrightwidth =.5pt cellheight=.23in } '
    ('~S={borderbottomcolor=cxD9D9D9 borderbottomwidth=.5pt
      bordertopcolor  =cxD9D9D9 bordertopwidth=.5pt
      borderrightcolor =black borderrightwidth =.5pt cellheight=.23in }Demographic'
      ('~S={borderbottomcolor=black borderbottomwidth=.5pt
        bordertopcolor  =cxD9D9D9 bordertopwidth=.5pt
        borderrightcolor =black borderrightwidth =.5pt
        cellheight=.23in }Category' var val
      )
    )
  )
  ('~S={borderbottomcolor=black borderbottomwidth=.5pt
    borderleftcolor  =black borderleftwidth  =.5pt }Extracted Factor'
    ( ('~S={borderbottomcolor=cxD9D9D9 borderbottomwidth=.5pt
      borderrightcolor =black borderrightwidth =.5pt
      cellheight=.23in } '
        ('~S={borderbottomcolor=black borderbottomwidth=.5pt
          borderleftcolor  =black borderleftwidth  =.5pt
          bordertopcolor  =cxD9D9D9 bordertopwidth=.5pt
          borderrightcolor =black borderrightwidth =.5pt
          cellheight=.23in }# number
        )
      )
    )
  )
  ('~S={ borderbottomcolor=black borderbottomwidth=.5pt
    borderleftcolor  =black borderleftwidth  =.5pt }Initial'
    ('~S={borderbottomcolor=black borderbottomwidth=.5pt
      borderleftcolor  =black borderleftwidth  =.5pt }Eigenvalue' eigenvalue
    )
  )
  ('~S={borderbottomcolor=black borderbottomwidth=.5pt
    borderleftcolor  =black borderleftwidth  =.5pt } Variance Accounted for'
    ('~S={borderbottomcolor=black borderbottomwidth=.5pt
      borderleftcolor  =black borderleftwidth  =.5pt
      borderrightcolor =black borderrightwidth =.5pt
      cellheight=.23in }%' proportion
    )
    ('~S={borderbottomcolor=black borderbottomwidth=.5pt
      borderleftcolor  =black borderleftwidth  =.5pt
      }Cumulative %' cumulative
    )
  )
)

```

```

);
)

DEFINE var / display '' center
  style(column)={cellheight=.23in cellwidth=.88in vjust=middle
    borderrightcolor=black borderrightwidth=.5pt};
DEFINE val / display '' center
  style(column)={cellheight=.23in cellwidth=.88in vjust=middle
    borderrightcolor=black borderrightwidth=.5pt};
DEFINE number / analysis '' center
  style(column)={cellheight=.23in cellwidth=.46in vjust=middle
    borderleftcolor =black borderleftwidth =.5pt
    borderrightcolor=black borderrightwidth=.5pt };
DEFINE eigenvalue / display '' center format=6.2
  style(column)={cellheight=.23in cellwidth=.88in vjust=middle
    borderleftcolor =black borderleftwidth =.5pt
    borderrightcolor=black borderrightwidth=.5pt };
DEFINE proportion / display '' center format=6.2
  style(column)={cellheight=.23in cellwidth=.62in vjust=middle
    borderleftcolor =black borderleftwidth =.5pt
    borderrightcolor=black borderrightwidth=.5pt };
DEFINE cumulative / display '' center format=6.2
  style(column)={cellheight=.23in cellwidth=1.2in vjust=middle
    borderleftcolor =black borderleftwidth =.5pt};
DEFINE rowline / noprint analysis;

COMPUTE rowline;
  if rowline.sum=1 then call define (_row_,'style',
    'style=[borderbottomcolor=black borderbottomwidth=.5pt]');
ENDCOMP;
COMPUTE number;
  if number.sum=1 then call define (_row_,'style',
    'style=[fontweight=bold]');
ENDCOMP;

RUN;

```