Advanced PROC REPORT: Traffic Lighting - Controlling Cell Attributes With Your Data

Arthur L. Carpenter
California Occidental Consultants

ABSTRACT

Color or shading is often used in a report table to attract the reader's eye to specific locations or items. The color can be fixed or it can be data dependent. Of course when the color or shading is dependent on the value presented in a given cell, manual construction of the table can be very tedious. Fortunately, with the advent of the Output Delivery System, a number of new capabilities have become available to the SAS® programmer. These include the ability to automatically control the color or shading of a particular cell of the table, and to even base that control on the data itself. The control of cell attributes, such as foreground and background color, is known as traffic lighting.

While traffic lighting can be performed on tables generated by TABULATE and PRINT as well, PROC REPORT has the more sophisticated capabilities. The CALL DEFINE statement is only available in the REPORT step's compute block, and the STYLE= option is more complex. This paper will discuss the use of user defined formats with the STYLE= option and the CALL DEFINE statement to control cell attributes. The discussion will include interactions with data summaries, computed variables, and various ODS destinations, such as RTF, PDF, and HTML .

KEYWORDS

Traffic Lighting, CALL DEFINE, STYLE=, Formats, ODS

INTRODUCTION

The term 'Traffic Lighting' refers to table items whose characteristics, such as background color, foreground color, font, and font size, change according to the value that is being displayed. Perhaps if a value is out of limits then we would like that cell to have a red background. This allows you to automatically highlight the values that you really want your reader to see.

The implementation is both ingenious and simple. We build custom formats using PROC FORMAT. These formats map the values of interest into style attributes. This makes changing the highlighted characteristics as easy as changing a format.

BUILDING TRAFFIC LIGHTING FORMATS

Traffic lighting depends on customized formats. There is nothing special about these formats other than the fact that they resolve not to a value that is to be displayed, but rather to an attribute that will be interpreted when the report is displayed.

The following PROC FORMAT will create two formats that can be used to control foreground and background colors. The formatted value (the item on the right of the equal sign) can be any of the style attribute values.

```
proc format;
   value cfore
      low - 21000     = 'white'
      21000< - 25000 = 'black'
      75000  - high   = 'white';
   value cback
```

```
         low - 21000    = 'red'
         21000< - 25000 = 'yellow'
         75000  - high  = 'green';
      run;
```

With these formats we would like to see values less than 21,000 to be displayed with white letters and a red background, while values over 75,000 will be displayed with a green background. These two example formats are used throughout the examples in this paper, and are assumed to exist even if the PROC FORMAT step is not actually shown in each example.

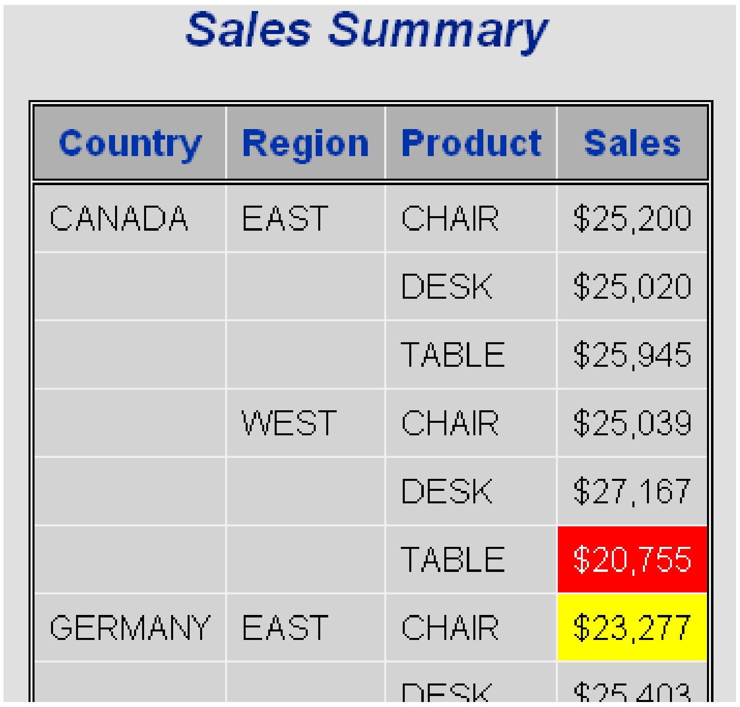
Of course even though these examples only change the foreground and background color, values for the other attributes can also be specified. You can associate font, font size, and other attribute values through the use of formats. Other clever usages have even included links and images that are dependent on the value that is displayed.

USING FORMATS WITH THE STYLE= OPTION

The report value that is to be displayed must be associated with a format and one way that this can be done is through the STYLE= option. In the following example the analysis variable ACTUAL has a FORMAT= option ❶ for the display of the values and a STYLE= option ❷ which utilizes the traffic lighting formats to control the foreground and background colors.

```
proc format;
   value cfore
      low    - 21000 = 'white'
      21000< - 25000 = 'black'
      >50000         = 'white';
   value cback
      low    - 21000 = 'red'
      21000< - 25000 = 'yellow'
      >50000         = 'green';
   run;

ods listing close;
ods html style=default
         path="&path\results"
         body='ch8_4_2a.html';

title1 'Sales Summary';
proc report data=sashelp.prdsale(where=(prodtype='OFFICE'))
      nowd;
   column country region product actual;
   define country / group;
   define region  / group;
   define product / group;
   define actual  / analysis sum
                    format=dollar8. ❶
                    'Sales'
                    style(column) = {background=cback. ❷
                                     foreground=cfore.};

   run;
ods html close;
```

A portion of the resulting table shows that values less than $25,000 are highlighted for quick recognition by the sales managers:



Sales Summary

Country	Region	Product	Sales
CANADA	EAST	CHAIR	$25,200
		DESK	$25,020
		TABLE	$25,945
	WEST	CHAIR	$25,039
		DESK	$27,167
		TABLE	🟥 $20,755
GERMANY	EAST	CHAIR	🟨 $23,277
		DESK	$25,403

Users will often try to apply the traffic lighting formats to all the data columns by placing the STYLE= option on the REPORT statement.

```
proc report data=sashelp.prdsale(where=(prodtype='OFFICE'))
      nowd
      style(column) = {background=cback.
                       foreground=cfore.};
```

While this approach can work to some degree, it can and usually does, cause some problems as well. This is because REPORT will attempt to apply the formats to all the columns including the grouping columns (REGION, COUNTRY, and PRODUCT) and in this case these are all character variables.

The report in the previous example was fairly straight forward. When we start adding summary lines, things can become more complex. If we add a BREAK and RBREAK statement to the REPORT step, a logical extension of what we did in the previous example, would be to add a STYLE(SUMMARY)= option to these two statements.

```
break after country / summarize suppress
                      style(summary) = {background=cback.
                                        foreground=cfore.};
rbreak after / summarize
               style(summary) = {background=cback.
                                 foreground=cfore.};
```

Unfortunately the timing is such that the execution of these statements and the resolution of the formats will **NOT** yield the desired results. When we want to provide traffic lighting for summary lines or even values under ACROSS variables, we will usually need to use the CALL DEFINE statement. Of course this statement can only be executed inside of a compute block, however, since we can assign a compute block for each column and summary line we have the ability to use the CALL DEFINE. This is discussed in the following section.

CONTROLLING TRAFFIC LIGHTING WITH CALL DEFINE

The CALL DEFINE routine can also be used in a compute block to create traffic lighting effects. The traffic lighting in the first example in the previous section could also have been accomplished with a CALL DEFINE. In the following code a compute block containing a CALL DEFINE statement has replaced the STYLE= option on the DEFINE ACTUAL statement.

```
ods listing close;
ods html style=default
         path="&path\results"
         body='ch8_4_3a.html';

title1 'Sales Summary';
proc report data=sashelp.prdsale(where=(prodtype='OFFICE'))
      nowd;
   column country region product actual;
   define country / group;
   define region  / group;
   define product / group;
   define actual  / analysis sum
                    format=dollar8.
                    'Sales';
   compute actual;
      call define(_col_,
                  'style',
                  'style = {background=cback.
                            foreground=cfore.}');
   endcomp;
   run;
ods html close;
```

This code which uses the same formats as were established earlier in this paper creates the same table as in the previous example. The compute block only exists to establish a location to place the CALL DEFINE statement. Certainly other executable compute block statements could also be placed in this compute block.

TRAFFIC LIGHTING IN THE PRESENCE OF COMPUTED VARIABLES AND SUMMARY LINES

Very often the best way to produce traffic lighting for reports that include computed variables and/or summary lines will be with a combination of CALL DEFINE statements and STYLE= options.

In the following examples we have complicated the previous examples by changing the PRODUCT variable to an ACROSS variable, computed a product total, and have requested country and report summarizations. We would like the traffic lighting formats to be applied to all of the numeric values including summary lines and the computed column.

There are several ways that we can do this and it is worth discussing the differences between the approaches. As was mentioned earlier, we cannot just use the STYLE= option on the BREAK, RBREAK, or DEFINE TOTALSALES statements.

For our first attempt, in the following example we take the approach of assigning the attributes by columns for all rows (detail and summary).

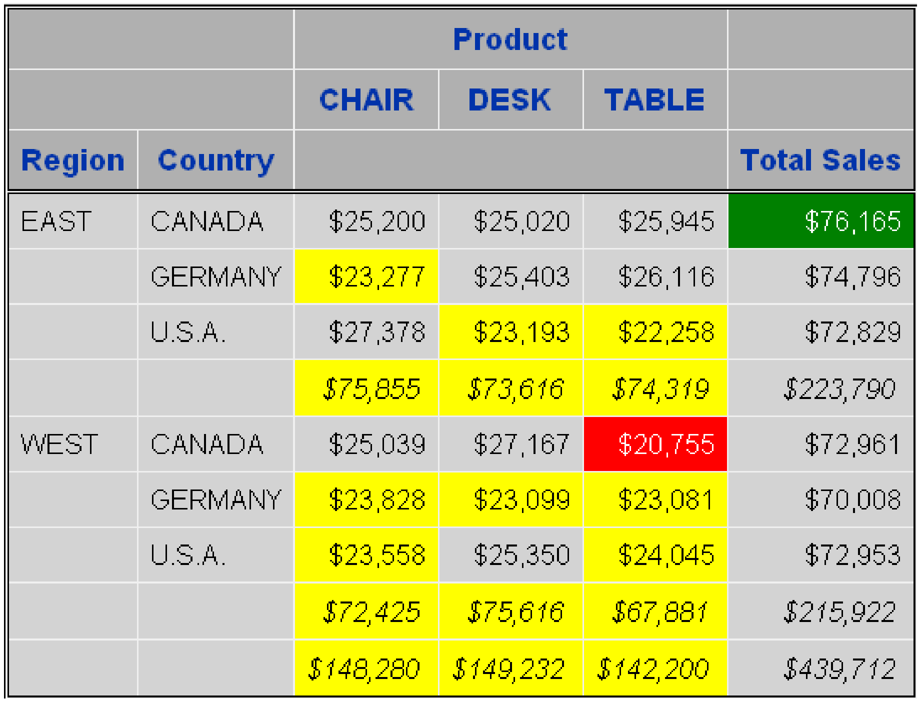
```
ods html style=default
         path="&path\results"
         body='ch8_4_4a.html';

proc report data=sashelp.prdsale(where=(prodtype='OFFICE'))
      nowd;
   column region country product,actual totalsales;
   define region  / group;
   define country / group;
   define product / across; ❶
   define actual  / analysis sum
                    format=dollar8.
                    'Sales'
                    style(column) = {background=cback. ❷
                                     foreground=cfore.};
   define totalsales / computed format=dollar10.
                       'Total Sales'
                       style(column) = {background=cback.❸
                                        foreground=cfore.};
   break after region / summarize suppress;
   rbreak after / summarize;

   compute totalsales;
      totalsales = sum(_c3_, _c4_, _c5_);
   endcomp;
   run;
ods html close;
```

❶ The PRODUCT is specified as an ACROSS variable (three columns).

❷ The three columns associated with the actual sales values are assigned these attributes. In the COLUMNS statement ACTUAL is nested under the across variable PRODUCTS.

❸ The same attributes are applied to the computed 'Total Sales' column.

Although the detail values have been formatted correctly the traffic lighting formats have been incorrectly applied to the summary lines.



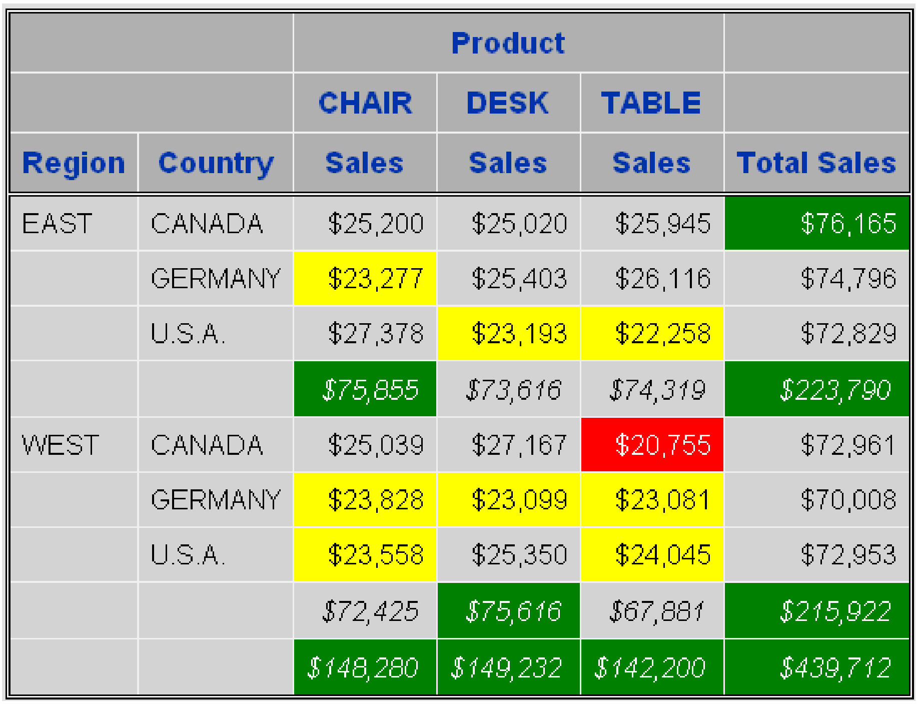
		Product			
		CHAIR	DESK	TABLE	
Region	Country				Total Sales
EAST	CANADA	$25,200	$25,020	$25,945	$76,165
	GERMANY	$23,277	$25,403	$26,116	$74,796
	U.S.A.	$27,378	$23,193	$22,258	$72,829
		$75,855	*$73,616*	*$74,319*	*$223,790*
WEST	CANADA	$25,039	$27,167	$20,755	$72,961
	GERMANY	$23,828	$23,099	$23,081	$70,008
	U.S.A.	$23,558	$25,350	$24,045	$72,953
		$72,425	*$75,616*	*$67,881*	*$215,922*
		$148,280	*$149,232*	*$142,200*	*$439,712*

In fairly simple tables, such as this, one way to apply the formatted style attributes to the summary lines is with the use of the summary component on the STYLE= option *e.g.* STYLE(SUMMARY)= ❹, as in the following.

```
ods html style=default
         path="&path\results"
         body='ch8_4_4b.html';
```

```
proc report data=sashelp.prdsale(where=(prodtype='OFFICE'))
      nowd
      style(summary)={background=cback.  ❹
                      foreground=cfore.};
   column region country product,actual totalsales;
   define region  / group;
   define country / group;
   define product / across;
   define actual  / analysis sum
                    format=dollar8.
                    'Sales'
                    style(column) = {background=cback.
                                     foreground=cfore.};
   define totalsales / computed format=dollar10.
                       'Total Sales'
                       style(column) = {background=cback.
                                        foreground=cfore.};
   break after region / summarize suppress;
   rbreak after / summarize;

   compute totalsales;
      totalsales = sum(_c3_, _c4_, _c5_);
   endcomp;
   run;
ods html close;
```

❹ The style attributes specified here will apply to all columns for all summary lines.



		Product			
		CHAIR	DESK	TABLE	
Region	Country	Sales	Sales	Sales	Total Sales
EAST	CANADA	$25,200	$25,020	$25,945	$76,165
	GERMANY	$23,277	$25,403	$26,116	$74,796
	U.S.A.	$27,378	$23,193	$22,258	$72,829
		$75,855	*$73,616*	*$74,319*	*$223,790*
WEST	CANADA	$25,039	$27,167	$20,755	$72,961
	GERMANY	$23,828	$23,099	$23,081	$70,008
	U.S.A.	$23,558	$25,350	$24,045	$72,953
		$72,425	*$75,616*	*$67,881*	*$215,922*
		$148,280	*$149,232*	*$142,200*	*$439,712*

This is a fairly general "brute force" solution. It may not work for all tables and it may be too simplistic for more complex situations. At the very least, control is lost by applying the same attributes to all of the different summary lines (of course this might be just what you want).

A similar table can be obtained by including the use of CALL DEFINE statements. This technique, which uses a combination of CALL DEFINE statements and STYLE= options, is far more extensible.

The following example will also produce the previous table, however the STYLE= option has been removed from the DEFINE TOTALSALES ❶, and is replaced with a CALL DEFINE statement in the compute block ❷. A good minimum rule of thumb is to use the CALL DEFINE when a compute block is present. In fact there is no real penalty for creating a compute block just in order to take advantage of the CALL DEFINE statement.

```
ods html style=default
         path="&path\results"
         body='ch8_4_4c.html';

proc report data=sashelp.prdsale(where=(prodtype='OFFICE'))
      nowd
      style(summary)={background=cback.
                      foreground=cfore.};
   column region country product,actual totalsales;
   define region   / group;
   define country / group;
   define product / across;
   define actual   / analysis sum
                    format=dollar8.
                    'Sales'
                    style(column) = {background=cback.
                                     foreground=cfore.};
   define totalsales / computed format=dollar10.
                       'Total Sales'; ❶

   break after region / summarize suppress;
   rbreak after / summarize;

   compute totalsales;
      totalsales = sum(_c3_, _c4_, _c5_);
      call define(_COL_,'style', 'style={background=cback. ❷
                                         foreground=cfore.}');
   endcomp;
   run;
ods html close;
```

DIFFERENTIATING BETWEEN COLUMNS

Because the CALL DEFINE can be conditionally executed and can reference individual columns, it provides us a great deal of flexibility that is not available through the STYLE= option. In the following example we would like the traffic lighting attributes to vary by values of the ACROSS variable (PRODUCT). This is accomplished by using direct column references in the CALL DEFINE statements.

To illustrate this level of control additional formats have been defined for the sales of chairs ❶, desks ❷, tables ❸, and for total sales ❹.

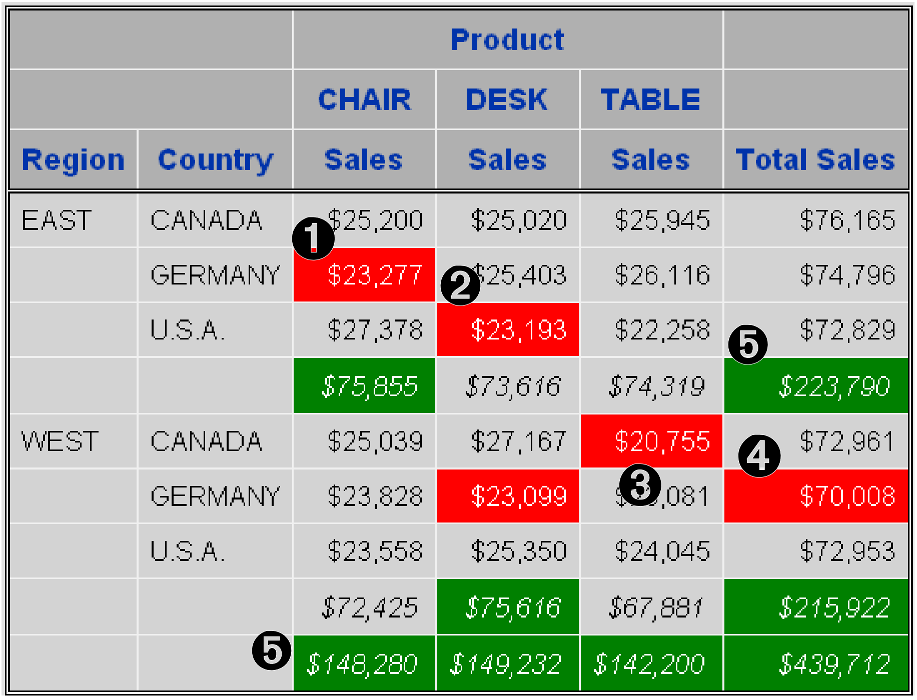
```
proc format;
   value cfore
      low    - 21000 = 'white'
      21000< - 25000 = 'black'
      75000  - high  = 'white';
   value cback
      low    - 21000 = 'red'
      21000< - 25000 = 'yellow'
      75000  - high  = 'green';
   value fchair ❶
      low    - 23500 = 'white';
   value bchair
      low    - 23500 = 'red';
   value fdesk ❷
      low    - 25000 = 'white';
   value bdesk
      low    - 25000 = 'red';
   value ftable ❸
      low    - 21000 = 'white';
   value btable
      low    - 21000 = 'red';
   value ftotal ❹
      low    - 72000 = 'white';
   value btotal
      low    - 72000 = 'red';
   run;

ods html style=default
         path="&path\results"
         body='ch8_4_5.html';

proc report data=sashelp.prdsale(where=(prodtype='OFFICE'))
      nowd
         style(summary)={background=cback.
                         foreground=cfore.};❺
   column region country product,actual totalsales;
   define region  / group;
   define country / group;
   define product / across;
   define actual  / analysis sum
                    format=dollar8.
                    'Sales';
   define totalsales / computed format=dollar10.
                       'Total Sales';

   break after region / summarize suppress;
   rbreak after / summarize;

   compute actual;
      if _break_ = ' ' then do; ❻
         call define('_C3_❼','style','style={background=bchair. ❶
                                           foreground=fchair.}');
         call define('_C4_','style','style={background=bdesk. ❷
                                           foreground=fdesk.}');
         call define('_C5_','style','style={background=btable. ❸
                                           foreground=ftable.}');
      end;
   endcomp;

   compute totalsales;
      totalsales = sum(_c3_, _c4_, _c5_);
      if _break_ = ' ' then do;
         call define(_COL_,'style', 'style={background=btotal. ❹
                                           foreground=ftotal.}');
      end;
   endcomp;
   run;
ods html close;
```

Formats are created and used to control the attributes of the sales of chairs ❶, desks ❷, tables ❸, and total sales ❹.

❺ In this example the summary lines are still being controlled with a STYLE(SUMMARY)= option. Traffic lighting for summary lines is discussed more in the following section.

❻ Only apply these CALL DEFINE statements to non-summary lines (`_break_ = ' '`).

❼ The specific column is named explicitly. This permits us to apply different formats to each type of sale item.



		Product			
		CHAIR	DESK	TABLE	
Region	Country	Sales	Sales	Sales	Total Sales
EAST	CANADA	❶ $25,200	$25,020	$25,945	$76,165
	GERMANY	$23,277	❷ $25,403	$26,116	$74,796
	U.S.A.	$27,378	$23,193	$22,258	❺ $72,829
		$75,855	*$73,616*	*$74,319*	*$223,790*
WEST	CANADA	$25,039	$27,167	$20,755	❹ $72,961
	GERMANY	$23,828	$23,099	$❸,081	$70,008
	U.S.A.	$23,558	$25,350	$24,045	$72,953
		$72,425	*$75,616*	*$67,881*	*$215,922*
	❺	*$148,280*	*$149,232*	*$142,200*	*$439,712*

DIFFERENTIATING BETWEEN COLUMNS ON GROUP SUMMARY ROWS

The previous example traffic lights all cells of all summary lines with the same format. When we need to fine tune the level of format application we have to be especially careful. The application of the attribute formats for cells that are determined through multiple compute blocks becomes complicated very quickly.

In the following example we want to apply the traffic lighting formats to the four columns of interest on the summary lines for the grouping variable REGION. To do this we have created four of new formats to apply to the region summaries (*x*REGN. ❶ is used for the summary of sales) and (*x*PROD. ❷ is used for the summary of TOTALSALES at the region summary level). Here *x* refers to either F or B (Foreground or Background).

```
proc format;
   value cfore
      low     - 21000 = 'white'
      21000< - 25000 = 'black'
      75000   - high  = 'white';
   value cback
      low     - 21000 = 'red'
      21000< - 25000 = 'yellow'
      75000   - high  = 'green';
   value fchair
      low     - 23500 = 'white';
   value bchair
      low     - 23500 = 'red';
   value fdesk
      low     - 25000 = 'white';
   value bdesk
      low     - 25000 = 'red';
   value ftable
```

```
      low    - 21000 = 'white';
   value btable
      low    - 21000 = 'red';
   value ftotal
      low    - 72000 = 'white';
   value btotal
      low    - 72000 = 'red';
   value fregn ❶
      low    - 73000 = 'white';
   value bregn
      low    - 73000 = 'red';
   value fprod ❷
      low    - 145000 = 'white';
   value bprod
      low    - 145000 = 'red';
   run;

ods html style=default
         path="&path\results"
         body='ch8_4_6.html';


proc report data=sashelp.prdsale(where=(prodtype='OFFICE'))
      nowd; ❸
   column region country product,actual totalsales;
   define region  / group;
   define country / group;
   define product / across;
   define actual  / analysis sum
                    format=dollar8.
                    'Sales';
   define totalsales / computed format=dollar10.
                       'Total Sales';

   break after region / summarize suppress;
   rbreak after / summarize;

   compute actual;
      if _break_ = ' ' then do;
         call define('_C3_','style','style={background=bchair.
                                            foreground=fchair.}');
         call define('_C4_','style','style={background=bdesk.
                                            foreground=fdesk.}');
         call define('_C5_','style','style={background=btable.
                                            foreground=ftable.}');
      end;
      else if _break_='REGION' then do; ❹
         call define('_C3_',❺ 'style','style={background=bregn. ❻
                                            foreground=fregn.}');
         call define('_C4_','style','style={background=bregn.
                                            foreground=fregn.}');
         call define('_C5_','style','style={background=bregn.
                                            foreground=fregn.}');
      end;
   endcomp;

   compute totalsales;
      totalsales = sum(_c3_, _c4_, _c5_);
      if _break_ = ' ' then do;
         call define(_COL_,'style', 'style={background=btotal.
                                            foreground=ftotal.}');
      end;
      else if _break_='REGION' then do; ❼
```

```
            call define(_col_,'style','style={background=cback.
                                                foreground=cfore.}');
         end;
      endcomp;
      run;
   ods html close;
```

❶ Specify formats for the REGION totals for sales of individual products.

❷ Specify formats for TOTALSALES (across products).

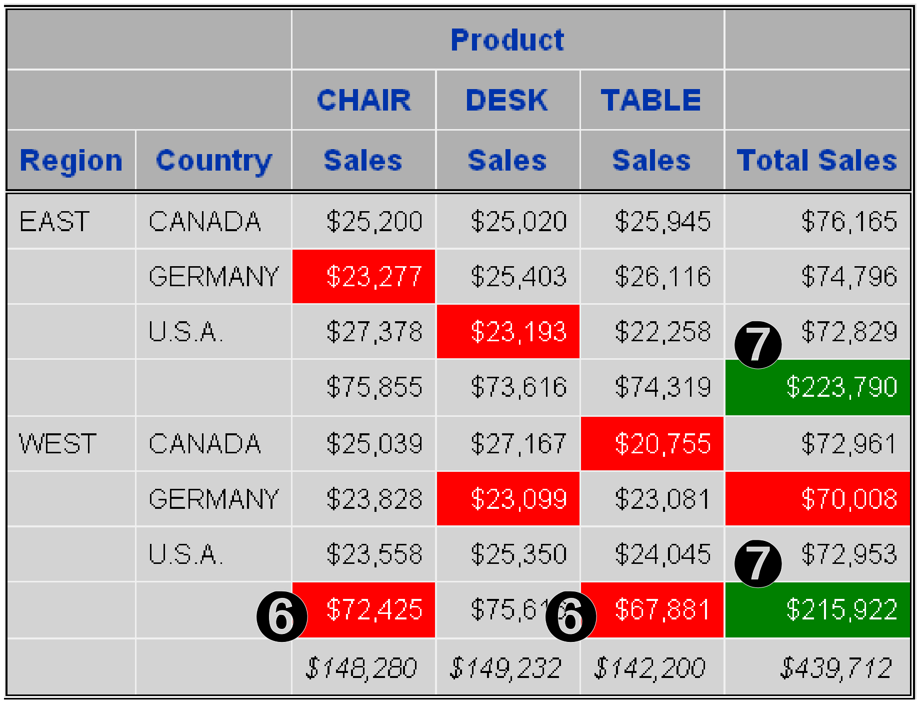
❸ The STYLE(SUMMARY)= option has been removed from the PROC statement. If it had been left in (as it was in the previous example), the CALL DEFINE statements would override the STYLE= option. However it has been my experience that this does not always work, as the timing for the declaration of the formats tend to conflict. My rule of thumb is to never allow a cell to be controlled by attributes derived by more than one format. Attributes specified directly (not through a format) do not seem to have this problem.

❹ A different set of CALL DEFINE statements is declared for the summary lines.

❺ The specific column is designated using direct addressing.

❻ The same format is applied to each of the three columns. The formats could have been different for each of the columns (as they are for the detail rows).

❼ This style is applied to the summary line for TOTAL SALES.



		Product			
		CHAIR	DESK	TABLE	
Region	Country	Sales	Sales	Sales	Total Sales
EAST	CANADA	$25,200	$25,020	$25,945	$76,165
	GERMANY	$23,277	$25,403	$26,116	$74,796
	U.S.A.	$27,378	$23,193	$22,258	❼ $72,829
		$75,855	$73,616	$74,319	$223,790
WEST	CANADA	$25,039	$27,167	$20,755	$72,961
	GERMANY	$23,828	$23,099	$23,081	$70,008
	U.S.A.	$23,558	$25,350	$24,045	❼ $72,953
		❻ $72,425	$75,616 ❻	$67,881	$215,922
		$148,280	*$149,232*	*$142,200*	*$439,712*

In this example all three columns for sales in the summary row (_c3_, _c4_, and _c5_) each receive the same format. When this happens, the code in the compute block could be simplified to a single CALL DEFINE statement. The COMPUTE ACTUAL block becomes:

```
      compute actual;
         if _break_ = ' ' then do;
            call define('_C3_','style','style={background=bchair.
                                                foreground=fchair.}');
            call define('_C4_','style','style={background=bdesk.
                                               foreground=fdesk.}');
            call define('_C5_','style','style={background=btable.
                                               foreground=ftable.}');
         end;
         else if _break_='REGION' then do;
            call define(_row_,'style','style={background=bregn.
                                              foreground=fregn.}');
         end;
      endcomp;
```

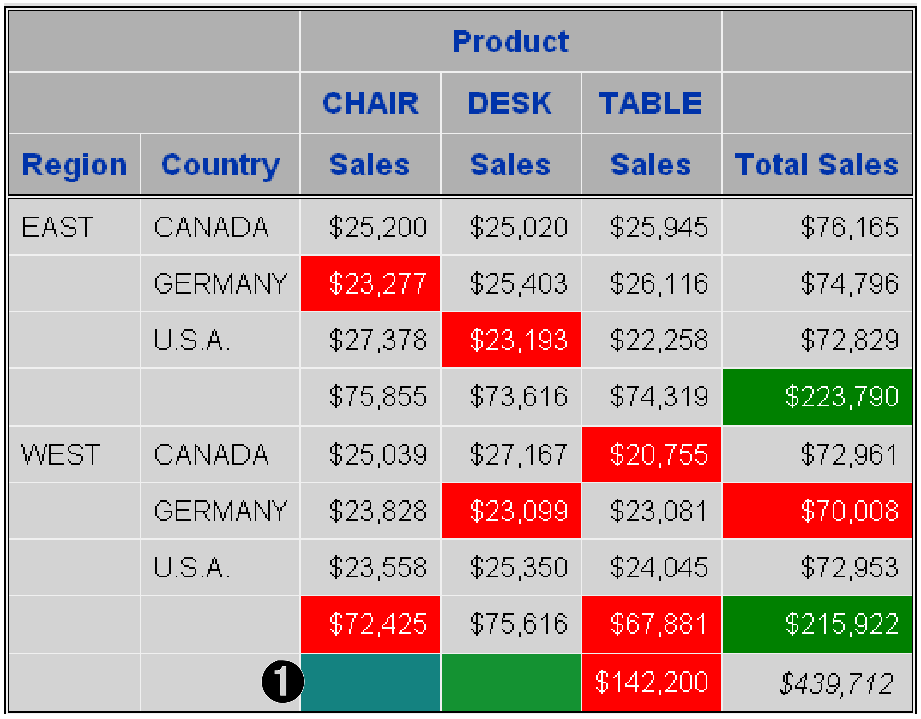
While this approach works in this example, I have not found this to always be the case. The problem, when it is a problem, seems to be caused by the use of _ROW_. Effectively, because we are using _ROW_, our formatted attributes would be applied to other columns than just _C3_, _C4_, and _C5_, and could cause a conflict in the TOTALSALES column.

TRAFFIC LIGHTING ON THE REPORT SUMMARY ROW

The specification of traffic lighting formats in the REPORT summary row is similar to that in the previous example. The difference is the addition of another ELSE IF statement used to detect the REPORT summary line (_BREAK_ = '_RBREAK_') when processing the report table. The COMPUTE ACTUAL block becomes:

```
compute actual;
   if _break_ = ' ' then do;
      call define('_C3_','style','style={background=bchair.
                                                foreground=fchair.}');
      call define('_C4_','style','style={background=bdesk.
                                                foreground=fdesk.}');
      call define('_C5_','style','style={background=btable.
                                                foreground=ftable.}');
   end;
   else if _break_='REGION' then do;
      call define('_C3_','style','style={background=bregn.
                                                foreground=fregn.}');
      call define('_C4_','style','style={background=bregn.
                                                foreground=fregn.}');
      call define('_C5_','style','style={background=bregn.
                                                foreground=fregn.}');
   end;
   else if _break_='_RBREAK_' then do;
      call define('_C3_','style','style={background=bprod.
                                                foreground=fprod.}');
      call define('_C4_','style','style={background=bprod.
                                                foreground=fprod.}');
      call define('_C5_','style','style={background=bprod.
                                                foreground=fprod.}');
   end;
endcomp;
```

Using the same format definitions for BPROD. and FPROD. that were used in the examples in the previous section, the following table is produced:

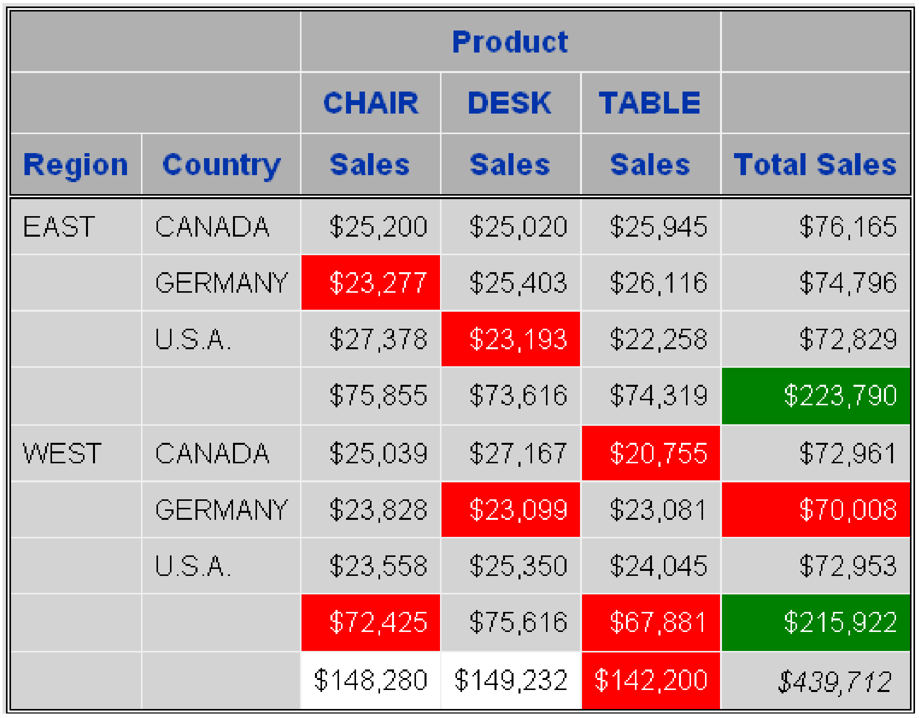


		Product			
		CHAIR	DESK	TABLE	
Region	Country	Sales	Sales	Sales	Total Sales
EAST	CANADA	$25,200	$25,020	$25,945	$76,165
	GERMANY	$23,277	$25,403	$26,116	$74,796
	U.S.A.	$27,378	$23,193	$22,258	$72,829
		$75,855	$73,616	$74,319	$223,790
WEST	CANADA	$25,039	$27,167	$20,755	$72,961
	GERMANY	$23,828	$23,099	$23,081	$70,008
	U.S.A.	$23,558	$25,350	$24,045	$72,953
		$72,425	$75,616	$67,881	$215,922
	❶	🟦	🟩	$142,200	*$439,712*

Clearly there is something wrong for the overall sales of CHAIRs and DESKs. This is a typical result when there is a timing issue with the determination of attributes that are established by using formats. In this case the two values that have been “colored over” ❶ did not receive a attribute from the format (their values are both over 145,000), but it was too late to use the default foreground and background colors. Here we can solve the problem through the use of the OTHER range specification which provides a place for all values in the format definition. Actually this is generally a good practice anyway. The format specifications become:

```
value fprod
   low     - 145000 = 'white'
   other            = 'black';
value bprod
   low     - 145000 = 'red'
   other            = 'white';
```

Using these format definitions the table becomes:



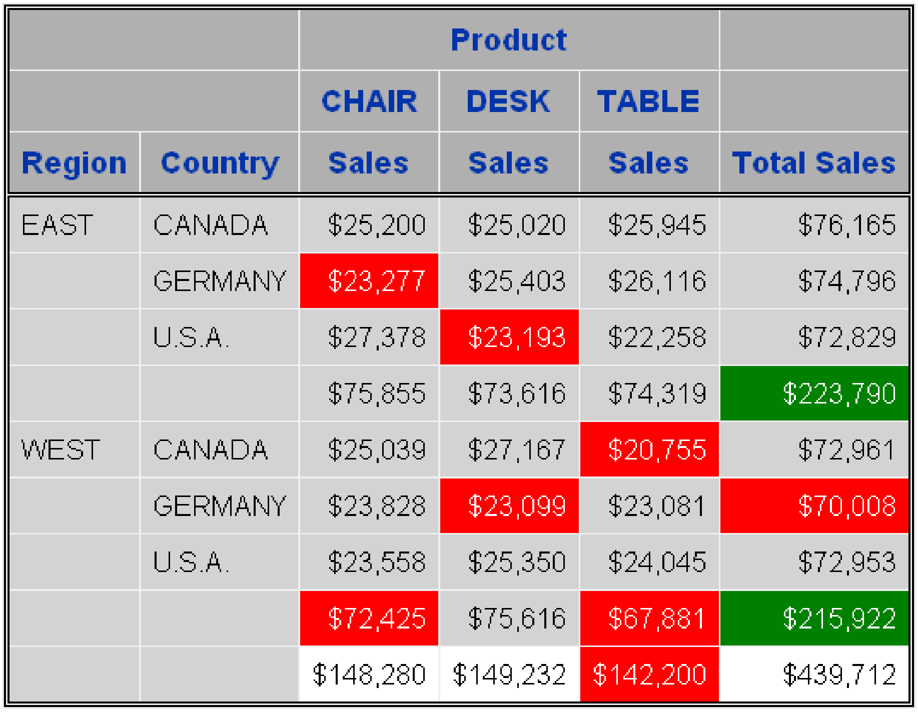
		Product			
		CHAIR	DESK	TABLE	
Region	Country	Sales	Sales	Sales	Total Sales
EAST	CANADA	$25,200	$25,020	$25,945	$76,165
	GERMANY	$23,277	$25,403	$26,116	$74,796
	U.S.A.	$27,378	$23,193	$22,258	$72,829
		$75,855	$73,616	$74,319	$223,790
WEST	CANADA	$25,039	$27,167	$20,755	$72,961
	GERMANY	$23,828	$23,099	$23,081	$70,008
	U.S.A.	$23,558	$25,350	$24,045	$72,953
		$72,425	$75,616	$67,881	$215,922
		$148,280	$149,232	$142,200	*$439,712*

To highlight what is happening, I used a background color (WHITE) that does not match the default background color for these cells. If you want the unformatted cells to 'blend in', the values for the OTHER specification should be the same as the default attributes for that ODS style.

As in the example in the previous section, we could consolidate the three CALL DEFINE statements into one. The new compute block becomes:

```
   compute actual;
      if _break_ = ' ' then do;
         call define('_C3_','style','style={background=bchair.
                                           foreground=fchair.}');
         call define('_C4_','style','style={background=bdesk.
                                           foreground=fdesk.}');
         call define('_C5_','style','style={background=btable.
                                           foreground=ftable.}');
      end;
      else if _break_='REGION' then do;
         call define(_row_,'style','style={background=bregn.
                                          foreground=fregn.}');
      end;
      else if _break_='_RBREAK_' then do;
         call define(_row_,'style','style={background=bprod.
                                          foreground=fprod.}');
      end;
   endcomp;
```

The resulting table is slightly different as the color attributes associated with the overall total, which were previously unformatted, now also receive the two traffic lighting formats. The table becomes:



		Product			
		CHAIR	DESK	TABLE	
Region	Country	Sales	Sales	Sales	Total Sales
EAST	CANADA	$25,200	$25,020	$25,945	$76,165
	GERMANY	$23,277	$25,403	$26,116	$74,796
	U.S.A.	$27,378	$23,193	$22,258	$72,829
		$75,855	$73,616	$74,319	$223,790
WEST	CANADA	$25,039	$27,167	$20,755	$72,961
	GERMANY	$23,828	$23,099	$23,081	$70,008
	U.S.A.	$23,558	$25,350	$24,045	$72,953
		$72,425	$75,616	$67,881	$215,922
		$148,280	$149,232	$142,200	$439,712

A FEW THINGS TO REMEMBER

There are a number of issues associated with the use of formats in the process of assigning style attribute values. Most of these are a result of timing issues. The problems are complex enough, and come up often enough, to warrant the summary that follows. Remember these are only issues if you are assigning style attributes using formats.

Both CALL DEFINE routines and STYLE= options can be used with traffic lighting formats, and in some cases they can be used interchangeably. Both can be used in the same step.

As a general rule, traffic lighting effects that are desired for a given cell, should be applied through a single CALL DEFINE routine or STYLE= option. When two different sources address the same cell, the resulting conflict can produce interesting, but unanticipated, results.

A CALL DEFINE routine in a compute block will take precedence over a corresponding STYLE= option.

I generally prefer the CALL DEFINE routine to the STYLE= option. The CALL DEFINE usually offers more control, especially when explicit column addressing is involved. However I often will make an exception if a STYLE= option does the trick, and the use of a CALL DEFINE would require an otherwise unnecessary compute block.

One caveat associated with the use of the CALL DEFINE routine is memory usage. For very large reports or for memory constrained environments, the STYLE= option may be preferred over CALL DEFINE as it tends to use less memory. This becomes even less of an issue in SAS 9.2.

It is not a bad idea to specify your traffic lighting formats to cover the full range. Not doing so allows the default style attributes to be displayed, however in some cases, especially for REPORT wide summary lines, timing issues prevent the default attributes from being correctly interpreted.

SUMMARY

The use of traffic lighting techniques has huge ramifications for users of the Output Delivery System. With the additional ability to control attributes such as foreground and background color through the use of formats, the coding for the table becomes independent of the data and the traffic lighting becomes data driven.

PROC REPORT has the capability of taking advantage of the traffic lighting capabilities of ODS through the use of the CALL DEFINE routine and the STYLE= option. Both are flexible and powerful.

In complex reports timing issues with regard to the resolution of formats require you to be careful with your choice of statements and how they are applied.

ABOUT THE AUTHOR

Art Carpenter's publications list includes three books, and numerous papers and posters presented at SUGI and other user group conferences. Art has been using SAS® since 1976 and has served in various leadership positions in local, regional, national, and international user groups. He is a SAS Certified Professional™ and through California Occidental Consultants he teaches SAS courses and provides contract SAS programming support nationwide.

AUTHOR CONTACT

Arthur L. Carpenter
California Occidental Consultants
10606 Ketch Circle
Anchorage, AK 99515

(907) 865-9163
art@caloxy.com
www.caloxy.com

REFERENCES

Portions of this paper were borrowed, with permission, from Arthur L. Carpenter's book on PROC REPORT. *Carpenter's Complete Guide to the SAS® Report Procedure*, Cary, NC: SAS Institute Inc.

Fan, Zizhong, 2005, "Make the Invisible Visible – A Case Study of Using ODS Inline Formatting Style", *Proceedings of the Thirtieth Annual SAS® Users Group International Conference*, Cary, NC: SAS Institute Inc., paper 042-30.

Hadden, Louise, 2006, "STOP! WAIT! GO!:See What Traffic-Lighting Can Do For You!", *Proceedings of the Thirty-first Annual SAS® Users Group International Conference*, Cary, NC: SAS Institute Inc., paper 142-31.

Haworth, Lauren E., 2001, "ODS for PRINT, REPORT and TABULATE", *Proceedings of the Twenty-sixth Annual SAS® Users Group International Conference*, Cary, NC: SAS Institute Inc., paper 003-26. Also published in the *Proceedings of the SouthEast SAS® Users Group Conference (SESUG 2001)*, Cary, NC: SAS Institute Inc., paper p-828.

TRADEMARK INFORMATION