Paper 189-27

PROC TABULATE: Doin’ It in Style!

Ray Pass, Ray Pass Consulting, Hartsdale, NY
Sandy McNeill, SAS, Cary, NC

Abstract

The advent of the SAS® Output Delivery System (ODS to its friends) has turned SAS reports from machine-generated, black & white monospace bores into people-produced, productive and reader-friendly information displays. One of the main principles underlying ODS is the use of Table and Style definitions (also known as Table and Style templates). Most procedures have a standard output layout structure and rely on their Table and Style definitions to govern the cosmetic or stylistic appearance of their tables. Certain procedures (REPORT, TABULATE, etc), however, by the very nature of their complete structural customizability, do not rely on fixed external table definitions. For these procedures, stylistic customizations are performed through the use of the STYLE option, an ODS concept which is integrated into the heart of the procedures' syntax. This presentation will demonstrate the use of STYLEs in the TABULATE procedure.

Introduction

Prior to Version 8 (actually Version 7) of the SAS System, the only form of output available from PROC TABULATE was the listing file in the Output Window. The output was produced in SAS Monospace font with form characters (usually dashes) used for overlining and underlining. This was the acceptable (and in fact the only) way to bring attention to summary or total lines. There was no way to highlight any of the rows, columns or cells of the output. As Version 8 was released, HTML output was gaining huge popularity as the choice of medium for sharing information – reports, documents, charts. In the HTML world, monospace, fixed fonts were no longer preferred and it was now possible to use proportional fonts, colors, different font sizes, bolding and italics to bring attention to areas of reports that needed more attention from the reader. Version 8 contained the first production release of the Output Delivery System (ODS). One of the main features of ODS is the ability to produce output from all BASE procedures in alternate formats (known in ODS as destinations). One of the original ODS destinations, in addition to the default LISTING destination, was HTML. Output sent to this destination was rendered as HTML-tagged output suitable for viewing in HTML browsers.

Most BASE procedures follow certain fairly rigid structural guidelines in terms of the overall layout of the results, and the design of the layout is fairly consistent from run to run. All UNIVARIATE output for example follows a basic blueprint. This was not, however, possible with certain reporting procedures (REPORT, TABULATE, PRINT, etc) because of the infinite amounts of final data layouts that could be created depending on many data factors including variables used and reporting statistics chosen, as well as other design considerations. Therefore, while standard codified aspects of most procedure output could be individually customized via ODS and its accompanying TEMPLATE procedure, this was not possible for the reporting procedures because of the lack of standard replicable design features. To compensate for this lack of individual customizability, a system of STYLE formatting was made available for use in REPORT and TABULATE coding (now available in PRINT as well) which provided the ability to individually customize almost all design aspects of the procedure output.

This presentation will illustrate some of the many different features of ODS STYLEs as implemented in PROC TABULATE. This will be done through a series of examples, each using the same source data set. This data set, a subset of SASHELP.PRDSAL3, contains fictitious actual and predicted sales for the years 1997 and 1998 for the states or provinces of three countries – Mexico, U.S.A., and Canada. The examples will start off very simply and then build upon each other by adding or changing features until the final example, which will be a culmination of these features. Let’s get started with the code needed to create the data set used throughout.

```
data tabhow;
   set sashelp.prdsal3(keep=country state
                            quarter year month
                            actual predict);
   where (   country='U.S.A.'
             and (   state='North Carolina'
                  or state='New York'
                  or state='California'))
          or country='Canada'
          or country='Mexico';
run;
```

MakeData Code

Example 1 – Plain old HTML

This first example creates the basic table that we will be working with throughout the rest of the paper. The code for the report is as follows, with comments after the code. The output for all examples can be found at the end of the paper.

```
title1 'Example 1 - Simple Tabulate HTML';
*---------------------------------------------;
ods listing close;
ods html file = "tabhow1.htm";
*---------------------------------------------;
proc tabulate data=tabhow;
   class country state year;
   var   actual predict;
   table country * (state='State/Province'
                    all='** Whole Country **'),
         year='Sales per Year' *
         (actual ='Actual'
          predict='Predicted') *
         sum=' ' * f=comma9.0
       / box='Country by State by Year' ;
run;
*-----------------------------------------------;
ods html close;
ods listing;
```

Example 1 Code

In addition to the actual and predicted sales values for each of the years for the countries, we also have an ALL row which captures the total actual and predicted sales values. The output is sent to the ODS HTML destination via two simple ODS statements:

1) **ODS HTML FILE = 'tabhow1.htm';** – this statement defines the HTML output file to which the output will be written,

2) **ODS HTML CLOSE;** - this statement closes the output file and is necessary before the output is available for browsing.

The ODS LISTING CLOSE; and ODS LISTING; statements, while not necessary for the functionality of the ODS HTML destination routing, are usually an excellent addition to all ODS coding. They simply turn off, and then turn back on, the default ODS LISTING destination to conserve resources.

A few things should be noticed as you look at the report output: 1) the report is rather plain, 2) it is difficult to discern the ALL rows from the rest of the rows, 3) it is easy to confuse one country's values with another . These concerns will be addressed in the following examples.

Example 2 – STYLE on the HTML statement

In this example we start working on the appearance of the report. The only thing we will do here is to change the overall style that is used for the output.

```
title  'Example 2 - Add Style to ODS HTML
Statement';
*-----------------------------------------------;
ods listing close;
ods html file  = "tabhow2.htm"
         style = sasweb;
*-----------------------------------------------;
proc tabulate data=tabhow;
   class country state year;
   var   actual predict;

   table country * (state='State/Province'
                    all='** Whole Country **'),
         year='Sales per Year' *
         (actual ='Actual'
          predict='Predicted') *
         sum=' ' * f=comma9.0
       / box ='Country by State by Year';
run;
*-----------------------------------------------;
ods html close;
ods listing;
```

Example 2 Code

The only difference between this example and Example 1 is the STYLE option on the HTML statement. When a STYLE option is used on a global ODS statement which opens a destination, the style specified (and its elements) are used until the destination is closed.

In the first example, we were actually using the DEFAULT style even though we did not explicitly code it. When using the HTML destination, this is the STYLE which is used by default.

In Example 2, we use the SASWEB style. This style corresponds to the SASWEB style template which is located in the STYLES directory of the first readable itemstore that contains this style template. As long as the style resides in the STYLES directory, you can omit specifying "STYLES." before the style name, since that is the default. It’s just like specifying/not specifying “WORK” as the default libref when referring to temporary SAS data sets. If this is unclear at the moment, don’t worry. It’s a base concept in ODS and TEMPLATES, but is not critical to where we are going.

Notice when comparing the output from Example 1 to that for Example 2 that the SASWEB style not only affects the color of different areas of the report, but also changes the font and font size that is used throughout the report. If you want stylistic changes to be in effect for many reports, the best place to put those changes is in a style template which you can then simply call from report to report. But once again, this is a bit peripheral to the topic at hand.

Example 3 – STYLE on the TABULATE statement

Now that we have seen how to apply a style to an entire report (and one that could be used for many reports), the next task is to see how we can make changes to colors, fonts, or other stylistic attributes for just one report. The method to use is the STYLE option, but not the same STYLE option as we saw on the ODS global statement in Example 2. The goal of the rest of this paper is to examine different ways that the STYLE option can be used within a specific PROC TABULATE. Let's take a look at the code for Example 3 and then we'll discuss the first use of the STYLE option, namely on the TABULATE statement.

```
title  'Example 3 - Add Style to TABULATE
Statement';
*-----------------------------------------------;
ods listing close;
ods html file  = "tabhow3.htm"
         style = sasweb;
*-----------------------------------------------;
proc tabulate data=tabhow
```

```
                  style=[background=beige];
   class country state year;
   var   actual predict;

   table country * (state='State/Province'
                    all='** Whole Country **'),
         year='Sales per Year' *
         (actual ='Actual'
          predict='Predicted') *
         sum=' ' * f=comma9.0
       / box ='Country by State by Year';
run;
*--------------------------------------------;
ods html close;
ods listing;
```

Example 3 Code

The STYLE= option has the following general syntax:

```
STYLE<(location(s)>=<style-element-name>
      <[attribute1 = attribute-value-1
        attribute2 = attribute-value-2
            .          .        .
        attribute3 = attribute-value-3]>
```

Note: You can use either square brackets ([and]) or braces ({ and }).

The optional LOCATION identifier tells where in the report the style should be applied. Since PROC TABULATE does not use LOCATION in the STYLE option, we won't discuss it here. PROC REPORT and PROC PRINT however do use this feature and this is discussed in their documentation chapters.

The STYLE-ELEMENT-NAME refers to the name of a style element from the current style template in use. This is beyond the scope of this paper, but if you want to learn more about style elements, the place to go is the SAS Procedures Guide, Version 8 (online or hardcopy.)

The last part(s) of the STYLE option are pairings of attribute names for those style attributes that you want to set, and the values that you want to assign to them. The PROC TEMPLATE documentation (in The Complete Guide to the SAS Output Delivery System, Version 8), as well as the aforementioned V8 SAS Procedures Guide, contain lists of attributes and their possible values.

When a FORMAT option is used on the TABULATE statement, the format specified is applied to the data cells, as opposed to row or column headers. This same rule also applies when a STYLE option resides on the TABULATE statement – the named style is applied to the data cells of the report.

On the TABULATE statement, we see the code STYLE=[BACKGROUND=BEIGE]. There is no style element name specified and only one style attribute and value. Since the STYLE option on the TABULATE statement affects data cells, the background color of data cells will be the color beige.

Example 4 – STYLE on CLASS and CLASSLEV statements

Now that we are familiar with the basic syntax and operation of the STYLE option, let's use this option with other PROC TABULATE statements. In this example, we will use the STYLE option on both the CLASS and CLASSLEV statements. Let's take a look at the code first.

```
title  'Example 4 - Add Style to CLASS and
CLASSLEV Statements';
*----------------------------------------------;
ods listing close;
ods html file  = "tabhow4.htm"
         style = sasweb;
*----------------------------------------------;
proc tabulate data=tabhow
              style=[background=beige];
   class    country state;
   class    year / style=[foreground=black];
   classlev year / style=[foreground=black];
   var      actual predict;

   table  country * (state='State/Province'
                     all='** Whole Country **'),
          year='Sales per Year' *
         (actual ='Actual'
          predict='Predicted') *
         sum=' ' *f=comma9.0
       / box ='Country by State by Year';
run;
*----------------------------------------------;
ods html close;
ods listing;
```

Example 4 Code

In this example we chose to have the column spanning header for the YEAR variable ('Sales Per Year') and the actual values of the YEAR variable (1997, 1998) displayed in black instead of the default white (this default comes from the style element HEADER in the style template that we are using which is STYLES.SASWEB). The first thing to understand is that we need to use the FOREGROUND attribute since we want to change the color of the text (we would use the BACKGROUND attribute to change the background color of the cell.) The second thing to figure out is where to place the STYLE options. Since we want to change the label of the YEAR variable, we place a STYLE option on the CLASS statement for YEAR. Notice that by using PROC TABULATE’s support of multiple CLASS statements for different categorizing variables, we can apply different styles (or none at all) to different class variables. In this example, COUNTRY and STATE are not given styles, and the text color for the YEAR variable label is set to black.

OK, so we’ve changed the style on the CLASS statement for the YEAR variable, but that only affects the label for the variable itself ('Sales Per Year'), and not the actual values of the variable, 1997 and 1998. To do this, we need to use the STYLE option on the CLASSLEV statement for YEAR. This CLASSLEV statement was created for just this purpose – to access the class level values for a particular class variable. Just as we can have multiple CLASS statements, we can also have multiple CLASSLEV statements to allow different class variables to be styled differently. In this example, however, we are only stylizing the class levels for the variable YEAR, so we have one CLASSLEV statement for that variable, with the same style option as we had on the CLASS statement for YEAR. Unlike CLASS statements which *must* be present for every categorizing variable in the run, CLASSLEV statements are only needed for those variables for which you wish to make variable-level style changes.

Example 5 - STYLE on the BOX option

One area that is slightly removed from the variables and their statistics in the output is the BOX area. This area can nevertheless be stylized. Here is the code:

```
title  'Example 5 - Add Style to BOX option on
TABLE Statement';
*----------------------------------------------;
ods listing close;
ods html file  = "tabhow5.htm"
         style = sasweb;
*----------------------------------------------;
proc tabulate data=tabhow
              style=[background=beige];
   class    country state;
   class    year / style=[foreground=black];
   classlev year / style=[foreground=black];
   var      actual predict;

   table  country * (state='State/Province'
                     all='** Whole Country **'),
          year='Sales per Year' *
          (actual ='Actual'
           predict='Predicted') *
          sum=' ' * f=comma9.0
       /  box=[label='Country by State by Year'
               style=[background=beige
                      foreground=black
                      font_size=3]];
run;
*----------------------------------------------;
ods html close;
ods listing;
```

Example 5 Code

Setting a style for the BOX area only slightly complicates the process of just setting a string to appear there. Instead of just specifying the string, as in BOX='string', you have to enclose the string (LABEL=) in brackets as well as the style. Remember that the STYLE= is enclosed in brackets, within the outer BOX brackets. In our example, we set a label as well as a style with three attributes: BACKGROUND, FOREGROUND and FONT_SIZE.

Example 6 - STYLE on VAR statements

OK, we've colored CLASS variable column headers with STYLE on CLASS and CLASSLEV statements (the same can obviously be done with row CLASS variables) and we've done the BOX area. Now we come to the rest of the table, the guts, the data cells. The task at hand in this example and the next is to differentially color the ACTUAL and PREDICT columns so that they can be more readily separated by eye when viewing the table. We'll do just the headers ('Actual', 'Predicted') in Example 6 and get to the data cells for these columns in Example 7. Here is the code for Example 6:

```
title  'Example 6 - Add Style to VAR Statement';
*----------------------------------------------;
ods listing close;
ods html file  = "tabhow6.htm"
         style = sasweb;
*----------------------------------------------;
proc tabulate data=tabhow
              style=[background=beige];
   class    country state;
   class    year / style=[foreground=black];
   classlev year / style=[foreground=black];
   var      actual  /
            style=[background=lightgreen
                   foreground=black];
   var      predict /
            style=[background=lightcyan
                   foreground=black];

   table  country * (state='State/Province'
                     all='** Whole Country **'),
          year='Sales per Year' *
          (actual ='Actual'
           predict='Predicted') *
          sum=' ' * f=comma9.0
       /  box=[label='Country by State by Year'
               style=[background=beige
                      foreground=black
                      font_size=3]];
run;
*----------------------------------------------;
ods html close;
ods listing;
```

Example 6 Code

Just as we can have separate CLASS and CLASSLEV statements for separate CLASS variables or sets of variables, we can also have separate VAR statements. The structure of the STYLE syntax on the VAR statement is just like that of the CLASS or CLASSLEV statement. In our example, we are giving a black foreground to both ACTUAL and PREDICT, but we are giving them different background colors, lightgreen for ACTUAL and lightcyan for PREDICT. This is the first step towards color-coding these columns. Let's go on to Example 7 and see the next step.

Example 7 – STYLE on TABLE statements

In Example 1, we saw that placing a STYLE option on the PROC TABULATE statement affects the style of *all* of the data cells. The task is now to style the cells differentially, depending on the VARs that are being displayed. But first, the code:

```
title  'Example 7 - Add Style to the TABLE
Statement';
*-----------------------------------------------;
ods listing close;
ods html file  = "tabhow7.htm"
         style = sasweb;
*-----------------------------------------------;
proc tabulate data=tabhow
              style=[background=beige];
   class    country state;
   class    year / style=[foreground=black];
   classlev year / style=[foreground=black];
   var      actual  /
            style=[background=lightgreen
                   foreground=black];
   var      predict /
            style=[background=lightcyan
            foreground=black];

   table  country * (state='State/Province'
                     all='** Whole Country **'),
          year='Sales per Year' *
          (actual='Actual' *
               [style=[background=lightgreen]]
           predict='Predicted' *
               [style=[background=lightcyan]]) *
          sum=' ' * f=comma9.0
       /  box=[label='Country by State by Year'
               style=[background=beige
                      foreground=black
                      font_size=3]];
run;
*-----------------------------------------------;
ods html close;
ods listing;
```

Example 7 Code

You add styles to individual variables by inserting STYLE code in the crossings for the individual variables. In the code above, the variable YEAR in the column dimension is crossed with the variables ACTUAL and PREDICT. Then the variable ACTUAL is crossed with [STYLE=[BACKGROUND=LIGHTGREEN]] and the variable PREDICT is crossed with [STYLE=[BACKGROUND=LIGHTCYAN]]. Then, both are crossed with SUM='‘*FORMAT=COMMA9.0.

We now have the entire columns color-coded, not only the headings. You can see how much easier it is to compare the ACTUAL or PREDICT values between the two years, 1997 and 1998.

Example 8 – User-defined formats as STYLE attribute values & PARENTing

We've seen that we can style column headers and data cells under the headers – now let's switch gears a bit and see what we can do with the rows. In these next two examples, we are going to first apply color bands to the row headers, and then carry these colors across the entire rows. Let's take a look at the code for Example 8:

```
title1 'Example 8';
title2 '- Add Color Bands to the Major Row
Variable (COUNTRY) Using Format';
title3 '- Add Color Bands to the Other Row
Variable (STATE) Using <PARENT>';
title4 '- Add Style to ALL Keyword Using
<PARENT>';
*-----------------------------------------------;
ods listing close;
*-----------------------------------------------;
proc format;
   value $ctryfmt 'U.S.A.'  = 'navyblue'
                  'Canada'  = 'CX23d8FF'
                  'Mexico'  = 'lightblue';
run;
*-----------------------------------------------;
ods html file  = "tabhow8.htm"
         style = sasweb;
*-----------------------------------------------;
proc tabulate data=tabhow
              style=[background=beige];
   class    country state;
   class    year    / style=[foreground=black];
   classlev country /
            style=[background=$ctryfmt.];
   classlev state   /
            style=<parent>[foreground=black];
   classlev year    / style=[foreground=black];
   var      actual  /
            style=[background=lightgreen
                   foreground=black];
   var      predict /
            style=[background=lightcyan
                   foreground=black];

   table  country * (state='State/Province'
                     all='** Whole Country **'),
          year='Sales per Year' *
          (actual='Actual' *
               [style=[background=lightgreen]]
            predict='Predicted' *
               [style=[background=lightcyan]]) *
          sum=' ' * f=comma9.0
       /  box=[label='Country by State by Year'
               style=[background=beige
                     foreground=black
                     font_size=3]];
   keyword all / style=<parent>;
run;
*-----------------------------------------------;
ods html close;
ods listing;
```

Example 8 Code

There are two new features of importance that we have introduced in this example. First of all, we are using a user-defined format as an attribute value in a STYLE attribute pairing. We defined the format $CTRYFMT. and then used it as the background color in the STYLE option on the CLASSLEV statement for COUNTRY. This is an efficient and flexible method which creates the potential for powerful data-dependent processing which can eliminate rigid hard-coding, but let's not go too far astray.

The other new feature introduced here is that of *parenting*. We now have the colors that we want set up for the different countries, and we would like to have the rest of each row be the same color as that of its country. Moving along to the right, the classlevel cell for each state is the next part of each row to be colorized. The question is, How do we assign a color to the classlevel cell for each state when we want it to be the same color as its parent country, but the country is obtaining its value from a user-defined format? The answer is the new <PARENT> syntax. Let's examine the STYLE option on the CLASSLEV statement for STATE in the code above, STYLE=<PARENT>[FOREGROUND=BLACK]. This STYLE option first sets the style for the classlevel cells of STATE to be the same as that of its parent, COUNTRY. It is said to *inherit* the style. There are a set of rules for TABULATE as to exactly where inheritance comes from, and these are outlined in the documentation. Once the base style attributes are in place through inheritance, they can be over-ridden, as is the case here with FOREGROUND. This attribute is set to BLACK here. Notice that we also used <PARENT> for the keyword ALL. But we are not done with rows yet.

Example 9 – <PARENT> on the TABLE statement

All that is left to do to our rows is to have them display with the same color as their countries and states. First the code:

```
title1 'Example 9 - Add <PARENT> to the
crossing';
title2 '          - Data Cells Nows Inherit from
their parent, STATE';
*-----------------------------------------------;
ods listing close;
*-----------------------------------------------;
proc format;
value $ctryfmt 'U.S.A.' = 'navyblue'
               'Canada' = 'CX23d8FF'
               'Mexico' = 'lightblue';
run;
*-----------------------------------------------;
ods html file  = "tabhow9.htm"
         style = sasweb;
*-----------------------------------------------;
proc tabulate data=tabhow
              style=[background=beige];
   class    country state;
   class    year    / style=[foreground=black];
   classlev country /
            style=[background=$ctryfmt.];
   classlev state   /
            style=<parent>[foreground=black];
   classlev year    / style=[foreground=black];
   var      actual  /
            style=[background=lightgreen
                   foreground=black];
   var      predict /
            style=[background=lightcyan
                   foreground=black];

   table  country=''*(state=''
                      all='** Whole Country **')
          * [style=<parent>],
          year='Sales per Year'
          * (actual ='Actual'
             predict='Predicted')
          * sum=' ' * f=comma9.0
       /  box=[label='Country by State by Year'
               style=[background=beige
                      foreground=black
                      font_size=3]];
   keyword all / style=<parent>;
run;
*-----------------------------------------------;
ods html close;
ods listing;
```

Example 9 Code

In order to get the rows to display with the same styles as their parents, which in this case are those of STATE, we had to change the inheritance of the data cells. First we added a STYLE to the crossing of the row dimension variables (COUNTRY * STATE), and used <PARENT> in that style. We also had to remove the styles in the table crossing for our column dimension variables (YEAR * (ACTUAL PREDICT)). Had we not done this, these column dimension styles would have taken precedence over the row dimension styles in the line of inheritance. As mentioned earlier, there are inheritance rules which do apply and which must be accounted for in creating the desired display. For good measure, we also blanked out the labels for COUNTRY and STATE, although this was not all necessary.

Example 10 – adding images and other features

In this next example, we add an image to each COUNTRY cell. We also make use of another powerful style attribute. Here is the code:

```
title1 'Example 10 - Add Images and other
Features';
*-----------------------------------------------;
ods listing close;
*-----------------------------------------------;
proc format;
   value $ctryfmt 'U.S.A.' = 'navyblue'
                  'Canada' = 'CX23d8FF'
                  'Mexico' = 'light blue';

   value $ctrybr  'U.S.A.' = '<BR>U.S.A.'
                  'Canada' = '<BR>Canada'
                  'Mexico' = '<BR>Mexico';

   value $ctryflg 'U.S.A.' = "USFlag2.gif"
                  'Canada' = "CanadaFlag2.gif"
                  'Mexico' = "MexicoFlag2.gif";
run;
*-----------------------------------------------;
ods html file  = "tabhow10.htm"
         style = sasweb;
*-----------------------------------------------;
proc tabulate data=tabhow
              style=[background=beige];
   class    country state;
   class    year    / style=[foreground=black];
   classlev country /
            style=[preimage=$ctryflg.
                   protectspecialchars=off
                   background=$ctryfmt.];
   classlev state   /
            style=<parent>[preimage=''
                           foreground=black];
   classlev year    / style=[foreground=black];
   var      actual  ;
   var      predict ;
   table  country=''*(state=''
                      all='** Whole Country **')
          * [style=<parent>],
          year='Sales per Year'
          * (actual ='Actual'
             predict='Predicted')
          * sum=' ' * f=comma9.0
       /  box=[label='Country by State by Year'
               style=[background=beige
                      foreground=black
                      font_size=3]];
   keyword all / style=<parent>[preimage=''];

   format country $ctrybr.;
run;
*-----------------------------------------------;
ods html close;
ods listing;
```

Example 10 Code

Up to now, we have used the style attributes FOREGROUND, BACKGROUND, and FONT_SIZE, but there are *many* more. The two new ones used here are PREIMAGE and PROTECTSPECIALCHARS. The PREIMAGE attribute allows you to insert an image (we are using gifs in this example) before the text in any cell. There is a matching POSTIMAGE attribute and the syntax/usage is the same. In this example, we are adding an image of the flag of each country into the cell preceding the country's name. The method we are using is the same as we did with the country background colors – we are using a user-defined format. This time, however, we are specifying images in a format ($CTRYFLG.), instead of colors. The resulting attribute on the CLASSLEV statement for the COUNTRY variable, PREIMAGE=$CTRYFLG. accomplishes the task.

If we left the attributes as above, we would have the flag image preceding the country name, all on one line in each cell. In order to get the name to wrap under the image, we use another technique, and another style attribute. We create another format ($CTRYBR.) which concatenates the HTML line break tag <BR> to the country name value. The name for Canada for example, will now be output as '<BR>Canada'. We then use the attribute pairing PROTECTSPECIALCHARS=OFF on the CLASSLEV statement for COUNTRY to tell SAS *not* to protect the characters < and > (& is also not protected with this pairing) and to treat them as HTML tags. The syntax is perhaps a little counterintuitive, but you'll get used to it.

So far, so good, but we're not through here yet. Remember that we are *parenting* the styles for COUNTRY into STATE via a CLASSLEV statement for STATE, and those for STATE into the data cells in the TABLE statement. If we stopped here, the preimages for COUNTRY would flow into *all* the STATE and data cells. In order to avoid this, we have to provide a null PREIMAGE attribute (PREIMAGE='') for STATE. We do not have to do the same in the TABLE statement because the null value for the data cells will flow through from their parent, STATE.

The last piece of work to be done here is to add the same null preimage value to the style for the ALL keyword, as this also inherits from COUNTRY.

Example 11 –Traffic Lighting

One very powerful method in creating information displays is the technique of automatically calling visual attention to values in a report if those values exceed or fall below certain established maximum or minimum values, or fall within pre-specified ranges. The values and ranges can either be pre-determined and hard-coded, or can be created in data-driven manners. This criteria-dependent highlighting is usually referred to as *traffic lighting*. This last example will demonstrate simple hard-coded traffic lighting with our data. We are going to first enhance the original data set to create a new variable to which we will then apply the technique. As is usual, first the code:

```
data tabhow2;
   set sashelp.prdsal3(keep=country state
                            quarter year month
                            actual predict);
   where(   country='U.S.A.'
            and (  state='North Carolina'
                 or state='New York'
                 or state='California'))
         or country='Canada'
         or country='Mexico';

   diffpc = (actual-predict);
run;
*-----------------------------------------------;
title  'Example 11 - Add Traffic Lighting';
*-----------------------------------------------;
ods listing close;
*-----------------------------------------------;
proc format;
   value $ctryfmt 'U.S.A.' = 'navyblue'
                  'Canada' = 'CX23d8FF'
                  'Mexico' = 'light blue';

   value difffmt  low - <-10  = 'red'
                  -10 - 0     = 'lightpink'
                  other       = 'beige';
run;
*-----------------------------------------------;
ods html file  = "tabhow11.htm"
         style = sasweb;
*-----------------------------------------------;
proc tabulate data=tabhow2
              style=[background=beige];
   class    country state;
   class    year    / style=[foreground=black];
   classlev year    / style=[foreground=black];
   var      actual  /
            style=[background=lightgreen
                   foreground=black];
   var      predict /
            style=[background=lightcyan
                   foreground=black];
   var      diffpc  /
            style=[background=lightyellow
```

```
                   foreground=black];

   table  country * (state='State/Province'
                     all='** Whole Country **')
          * [style=<parent>],
          year='Sales per Year'
          * (actual='Actual'
             * sum='' * f=comma9.0
             * [style=[background=lightgreen]]
             predict='Predicted'
             * sum='' * f=comma9.0
             * [style=[background=lightcyan]]
             diffpc ='% Difference'
             * pctsum<predict>='' * f=7.2
             * [style=[background=difffmt.]])
       /  box=[label='Country by State by Year'
               style=[background=beige
                      foreground=black
                      font_size=3]];
run;
*-----------------------------------------------;
ods html close;
ods listing;
```

Example 11 Code

Traffic lighting is easily achieved in PROC TABULATE by creating user-defined formats and then by using these formats as attribute values in STYLE attribute pairings. This is the exact same method that we used for applying different color backgrounds to the different countries. This time, however, instead of the formats having individual values (such as the country names), we use ranges of values and then associate a color name with each range. We can then use the formats as BACKGROUND attributes.

Before we run the TABULATE, we will create an alternate to the original TABHOW dataset. This time, we create a new variable DIFF, which is simply the difference between predicted and actual values. This will be the basis for the variable to which we will apply the traffic lighting in the output. In the TABULATE code, we are actually going to display the percentage of this calculated value DIFF to the PREDICT value by using a PCTSUM statistic. We want to be able to visually recognize major and minor negative percent differences. In order to highlight these values, we create a new format, DIFFFMT.. This equates the range of LOW (the lowest number found in the data) to LESS THAN –10 with the color RED, and the range of -10 to 0 with the color LIGHTPINK. All other numbers are equated with the color BEIGE. This is the format that we will use in the output table with the statistic based on DIFF.

Since we will be displaying three different variables, ACTUAL, PREDICT and DIFF, with different output formats and with different STYLES, the TABLE statement has to deal with each of them separately. ACTUAL and PREDICT are not changed from previous examples – they will be displayed using a SUM statistic, a COMMA9.0 format and a fixed background color (LIGHTGREEN for ACTUAL and LIGHTCYAN for PREDICT.) The new variable DIFF, however, will be displayed using a PCTSUM statistic with PREDICT as the denominator, an output format of 7.2 and the newly created output format DIFFFMT. as the attribute value for the BACKGROUND attribute. Since the combinations of label, output format and STYLE is different for each variable, each one gets its own fully displayed crossing.

One warning with using formats and traffic lighting: you must make sure that the entire range of values is covered in your format statement. If some values are not covered, you may see unpredictable results with the style attribute for which you are using the format. In this example, we cover all possible values by using the OTHER parameter in building our output format.

Conclusion

PROC TABULATE has been around since the mid eighties and has proven to be a powerhouse in the field of summary data reporting. The advent of the Output Delivery System with Version 8 has greatly enhanced the ability to create data reports from most procedures that are visually compelling as well as data-rich. However, the marriage between ODS and TABULATE has not been as automatic as it has been with other, fixed-layout procedures. This is fully understandable because of the ability of TABULATE to produce an infinite amount of output data layouts. However, by using the STYLE option and its myriad of formatting features in your PROC TABULATE code, you can achieve the same amount of power and elegance as the other procedures, if not more so.

Acknowledgements

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

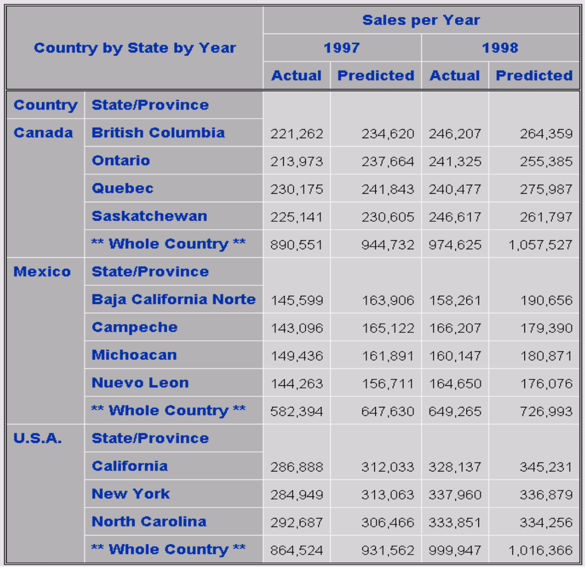
Author Contact Information

The authors of this paper can be contacted as follows:

Ray Pass
Ray Pass Consulting
5 Sinclair Place
Hartsdale, NY 10530
Voice: (914) 693-5553
eFax: (914) 206-3780
e-mail: raypass@att.net

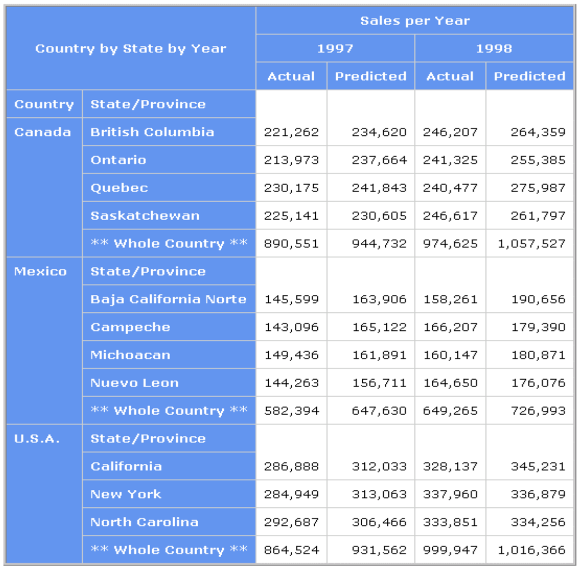
Sandy McNeill
SAS Institute
SAS Campus Drive
Cary, NC
Voice: (919) 531-5453
Fax: (919) 677-4444
e-mail: sandy.mcneill@sas.com

Example 1 - Simple Tabulate HTML



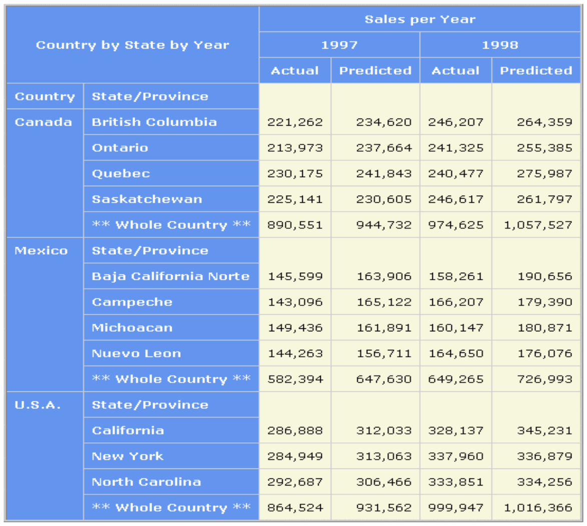
Country by State by Year		Sales per Year			
		1997		1998	
		Actual	Predicted	Actual	Predicted
Country	State/Province				
Canada	British Columbia	221,262	234,620	246,207	264,359
	Ontario	213,973	237,664	241,325	255,385
	Quebec	230,175	241,843	240,477	275,987
	Saskatchewan	225,141	230,605	246,617	261,797
	** Whole Country **	890,551	944,732	974,625	1,057,527
Mexico	State/Province				
	Baja California Norte	145,599	163,906	158,261	190,656
	Campeche	143,096	165,122	166,207	179,390
	Michoacan	149,436	161,891	160,147	180,871
	Nuevo Leon	144,263	156,711	164,650	176,076
	** Whole Country **	582,394	647,630	649,265	726,993
U.S.A.	State/Province				
	California	286,888	312,033	328,137	345,231
	New York	284,949	313,063	337,960	336,879
	North Carolina	292,687	306,466	333,851	334,256
	** Whole Country **	864,524	931,562	999,947	1,016,366

Example 2 - Add Style to ODS HTML Statement



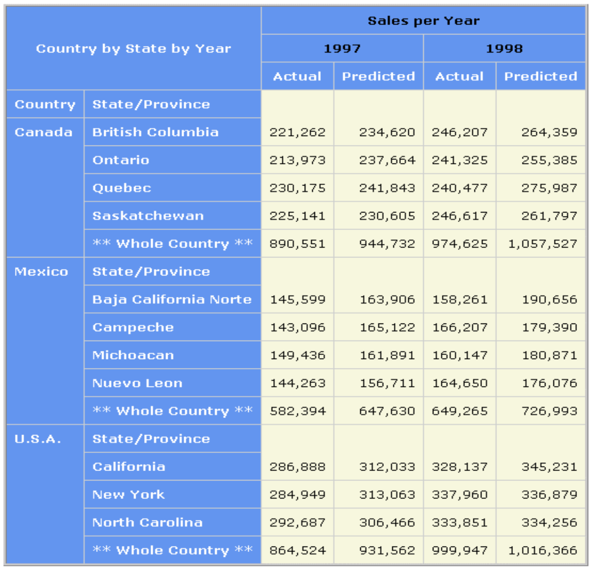
Country by State by Year		Sales per Year			
		1997		1998	
		Actual	Predicted	Actual	Predicted
Country	State/Province				
Canada	British Columbia	221,262	234,620	246,207	264,359
	Ontario	213,973	237,664	241,325	255,385
	Quebec	230,175	241,843	240,477	275,987
	Saskatchewan	225,141	230,605	246,617	261,797
	** Whole Country **	890,551	944,732	974,625	1,057,527
Mexico	State/Province				
	Baja California Norte	145,599	163,906	158,261	190,656
	Campeche	143,096	165,122	166,207	179,390
	Michoacan	149,436	161,891	160,147	180,871
	Nuevo Leon	144,263	156,711	164,650	176,076
	** Whole Country **	582,394	647,630	649,265	726,993
U.S.A.	State/Province				
	California	286,888	312,033	328,137	345,231
	New York	284,949	313,063	337,960	336,879
	North Carolina	292,687	306,466	333,851	334,256
	** Whole Country **	864,524	931,562	999,947	1,016,366

Example 3 - Add Style to TABULATE Statement



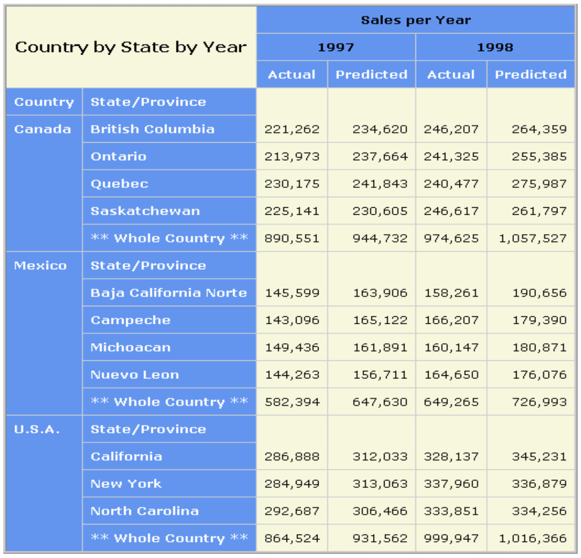
Country by State by Year		Sales per Year			
		1997		1998	
		Actual	Predicted	Actual	Predicted
Country	State/Province				
Canada	British Columbia	221,262	234,620	246,207	264,359
	Ontario	213,973	237,664	241,325	255,385
	Quebec	230,175	241,843	240,477	275,987
	Saskatchewan	225,141	230,605	246,617	261,797
	** Whole Country **	890,551	944,732	974,625	1,057,527
Mexico	State/Province				
	Baja California Norte	145,599	163,906	158,261	190,656
	Campeche	143,096	165,122	166,207	179,390
	Michoacan	149,436	161,891	160,147	180,871
	Nuevo Leon	144,263	156,711	164,650	176,076
	** Whole Country **	582,394	647,630	649,265	726,993
U.S.A.	State/Province				
	California	286,888	312,033	328,137	345,231
	New York	284,949	313,063	337,960	336,879
	North Carolina	292,687	306,466	333,851	334,256
	** Whole Country **	864,524	931,562	999,947	1,016,366

Example 4 - Add Style to CLASS and CLASSLEV Statements



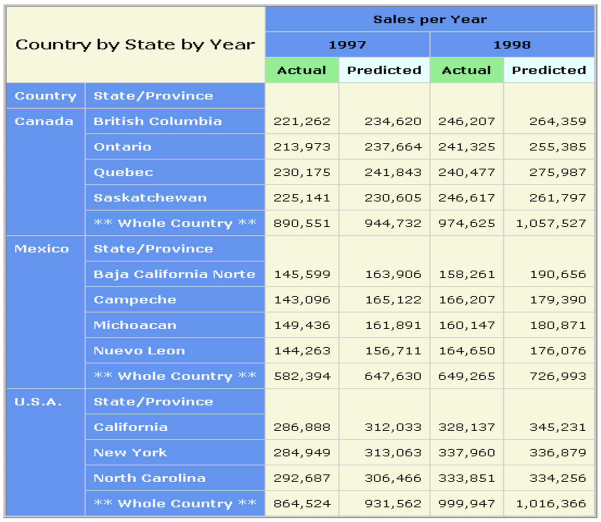
Country by State by Year		Sales per Year			
		1997		1998	
		Actual	Predicted	Actual	Predicted
Country	State/Province				
Canada	British Columbia	221,262	234,620	246,207	264,359
	Ontario	213,973	237,664	241,325	255,385
	Quebec	230,175	241,843	240,477	275,987
	Saskatchewan	225,141	230,605	246,617	261,797
	** Whole Country **	890,551	944,732	974,625	1,057,527
Mexico	State/Province				
	Baja California Norte	145,599	163,906	158,261	190,656
	Campeche	143,096	165,122	166,207	179,390
	Michoacan	149,436	161,891	160,147	180,871
	Nuevo Leon	144,263	156,711	164,650	176,076
	** Whole Country **	582,394	647,630	649,265	726,993
U.S.A.	State/Province				
	California	286,888	312,033	328,137	345,231
	New York	284,949	313,063	337,960	336,879
	North Carolina	292,687	306,466	333,851	334,256
	** Whole Country **	864,524	931,562	999,947	1,016,366

Example 5 - Add Style to BOX option on TABLE Statement



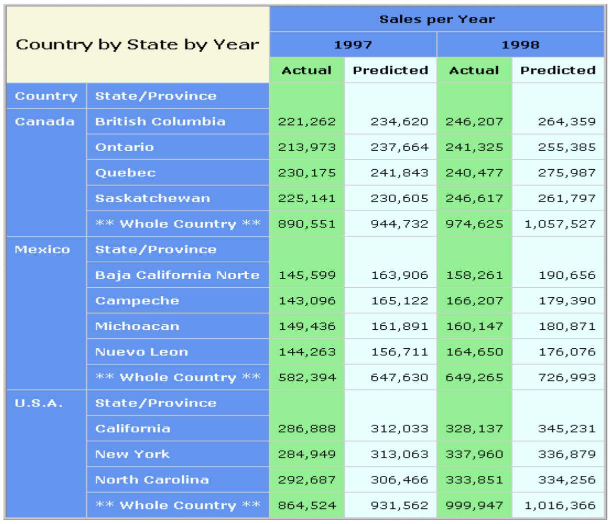
Country by State by Year		Sales per Year			
		1997		1998	
		Actual	Predicted	Actual	Predicted
Country	State/Province				
Canada	British Columbia	221,262	234,620	246,207	264,359
	Ontario	213,973	237,664	241,325	255,385
	Quebec	230,175	241,843	240,477	275,987
	Saskatchewan	225,141	230,605	246,617	261,797
	** Whole Country **	890,551	944,732	974,625	1,057,527
Mexico	State/Province				
	Baja California Norte	145,599	163,906	158,261	190,656
	Campeche	143,096	165,122	166,207	179,390
	Michoacan	149,436	161,891	160,147	180,871
	Nuevo Leon	144,263	156,711	164,650	176,076
	** Whole Country **	582,394	647,630	649,265	726,993
U.S.A.	State/Province				
	California	286,888	312,033	328,137	345,231
	New York	284,949	313,063	337,960	336,879
	North Carolina	292,687	306,466	333,851	334,256
	** Whole Country **	864,524	931,562	999,947	1,016,366

Example 6 - Add Style to VAR Statement



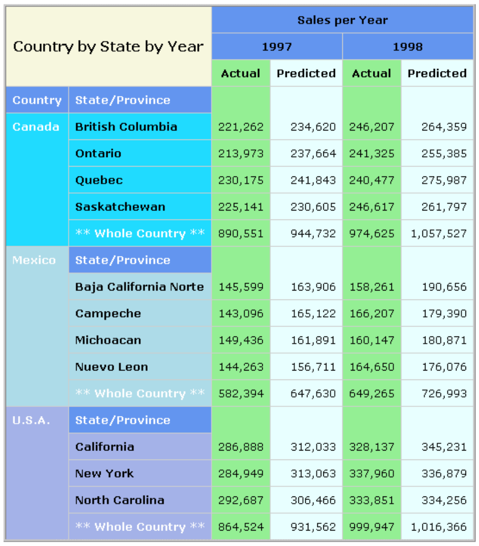
Country by State by Year		Sales per Year			
		1997		1998	
		Actual	Predicted	Actual	Predicted
Country	State/Province				
Canada	British Columbia	221,262	234,620	246,207	264,359
	Ontario	213,973	237,664	241,325	255,385
	Quebec	230,175	241,843	240,477	275,987
	Saskatchewan	225,141	230,605	246,617	261,797
	** Whole Country **	890,551	944,732	974,625	1,057,527
Mexico	State/Province				
	Baja California Norte	145,599	163,906	158,261	190,656
	Campeche	143,096	165,122	166,207	179,390
	Michoacan	149,436	161,891	160,147	180,871
	Nuevo Leon	144,263	156,711	164,650	176,076
	** Whole Country **	582,394	647,630	649,265	726,993
U.S.A.	State/Province				
	California	286,888	312,033	328,137	345,231
	New York	284,949	313,063	337,960	336,879
	North Carolina	292,687	306,466	333,851	334,256
	** Whole Country **	864,524	931,562	999,947	1,016,366

Example 7 - Add Style to the TABLE Statement



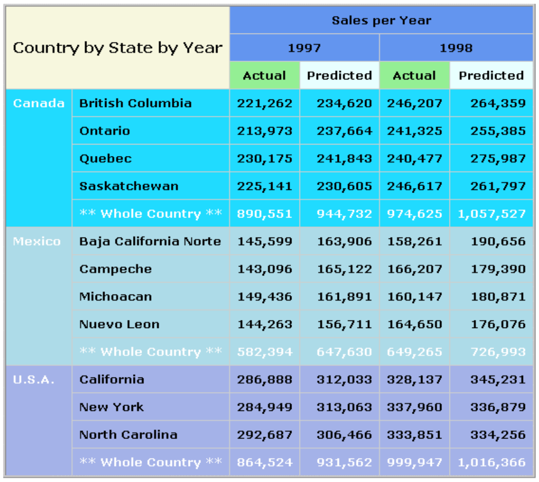
Country by State by Year		Sales per Year			
		1997		1998	
		Actual	Predicted	Actual	Predicted
Country	State/Province				
Canada	British Columbia	221,262	234,620	246,207	264,359
	Ontario	213,973	237,664	241,325	255,385
	Quebec	230,175	241,843	240,477	275,987
	Saskatchewan	225,141	230,605	246,617	261,797
	** Whole Country **	890,551	944,732	974,625	1,057,527
Mexico	State/Province				
	Baja California Norte	145,599	163,906	158,261	190,656
	Campeche	143,096	165,122	166,207	179,390
	Michoacan	149,436	161,891	160,147	180,871
	Nuevo Leon	144,263	156,711	164,650	176,076
	** Whole Country **	582,394	647,630	649,265	726,993
U.S.A.	State/Province				
	California	286,888	312,033	328,137	345,231
	New York	284,949	313,063	337,960	336,879
	North Carolina	292,687	306,466	333,851	334,256
	** Whole Country **	864,524	931,562	999,947	1,016,366

Example 8
- Add Color Bands to the Major Row Variable (COUNTRY) Using Format
- Add Color Bands to the Other Row Variable (STATE) Using <PARENT>
- Add Style to ALL Keyword Using <PARENT>



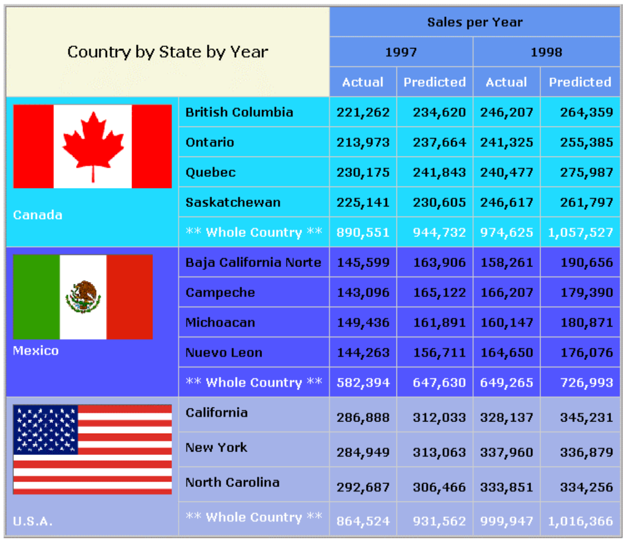
Country by State by Year		Sales per Year			
		1997		1998	
		Actual	Predicted	Actual	Predicted
Country	State/Province				
Canada	British Columbia	221,262	234,620	246,207	264,359
	Ontario	213,973	237,664	241,325	255,385
	Quebec	230,175	241,843	240,477	275,987
	Saskatchewan	225,141	230,605	246,617	261,797
	** Whole Country **	890,551	944,732	974,625	1,057,527
Mexico	State/Province				
	Baja California Norte	145,599	163,906	158,261	190,656
	Campeche	143,096	165,122	166,207	179,390
	Michoacan	149,436	161,891	160,147	180,871
	Nuevo Leon	144,263	156,711	164,650	176,076
	** Whole Country **	582,394	647,630	649,265	726,993
U.S.A.	State/Province				
	California	286,888	312,033	328,137	345,231
	New York	284,949	313,063	337,960	336,879
	North Carolina	292,687	306,466	333,851	334,256
	** Whole Country **	864,524	931,562	999,947	1,016,366

Example 9 - Add <PARENT> to the crossing
- Data Cells Nows Inherit from their parent, STATE



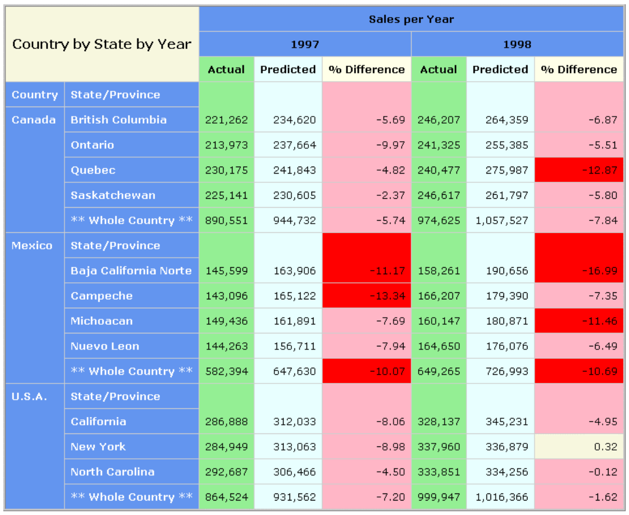
Country by State by Year		Sales per Year			
		1997		1998	
		Actual	Predicted	Actual	Predicted
Canada	British Columbia	221,262	234,620	246,207	264,359
	Ontario	213,973	237,664	241,325	255,385
	Quebec	230,175	241,843	240,477	275,987
	Saskatchewan	225,141	230,605	246,617	261,797
	** Whole Country **	890,551	944,732	974,625	1,057,527
Mexico	Baja California Norte	145,599	163,906	158,261	190,656
	Campeche	143,096	165,122	166,207	179,390
	Michoacan	149,436	161,891	160,147	180,871
	Nuevo Leon	144,263	156,711	164,650	176,076
	** Whole Country **	582,394	647,630	649,265	726,993
U.S.A.	California	286,888	312,033	328,137	345,231
	New York	284,949	313,063	337,960	336,879
	North Carolina	292,687	306,466	333,851	334,256
	** Whole Country **	864,524	931,562	999,947	1,016,366

Example 10 - Add Images and other Features



Country by State by Year		Sales per Year			
		1997		1998	
		Actual	Predicted	Actual	Predicted
Canada	British Columbia	221,262	234,620	246,207	264,359
	Ontario	213,973	237,664	241,325	255,385
	Quebec	230,175	241,843	240,477	275,987
	Saskatchewan	225,141	230,605	246,617	261,797
	** Whole Country **	890,551	944,732	974,625	1,057,527
Mexico	Baja California Norte	145,599	163,906	158,261	190,656
	Campeche	143,096	165,122	166,207	179,390
	Michoacan	149,436	161,891	160,147	180,871
	Nuevo Leon	144,263	156,711	164,650	176,076
	** Whole Country **	582,394	647,630	649,265	726,993
U.S.A.	California	286,888	312,033	328,137	345,231
	New York	284,949	313,063	337,960	336,879
	North Carolina	292,687	306,466	333,851	334,256
	** Whole Country **	864,524	931,562	999,947	1,016,366

Example 11 - Add Traffic Lighting



Country by State by Year		Sales per Year					
		1997			1998		
		Actual	Predicted	% Difference	Actual	Predicted	% Difference
Country	State/Province						
Canada	British Columbia	221,262	234,620	-5.69	246,207	264,359	-6.87
	Ontario	213,973	237,664	-9.97	241,325	255,385	-5.51
	Quebec	230,175	241,843	-4.82	240,477	275,987	-12.87
	Saskatchewan	225,141	230,605	-2.37	246,617	261,797	-5.80
	** Whole Country **	890,551	944,732	-5.74	974,625	1,057,527	-7.84
Mexico	State/Province						
	Baja California Norte	145,599	163,906	-11.17	158,261	190,656	-16.99
	Campeche	143,096	165,122	-13.34	166,207	179,390	-7.35
	Michoacan	149,436	161,891	-7.69	160,147	180,871	-11.46
	Nuevo Leon	144,263	156,711	-7.94	164,650	176,076	-6.49
	** Whole Country **	582,394	647,630	-10.07	649,265	726,993	-10.69
U.S.A.	State/Province						
	California	286,888	312,033	-8.06	328,137	345,231	-4.95
	New York	284,949	313,063	-8.98	337,960	336,879	0.32
	North Carolina	292,687	306,466	-4.50	333,851	334,256	-0.12
	** Whole Country **	864,524	931,562	-7.20	999,947	1,016,366	-1.62