PharmaSUG2011 – Paper HW07

An Introduction to SAS/GRAPH
or
Quick Tricks with the GPLOT and GCHART Procedures And the Annotate Facility

Ben Cochran, The Bedford Group, Raleigh, NC

ABSTRACT:
The power and flexibility of SAS/GRAPH software enables the user to produce high quality graphs, charts, and maps. With all this strength and flexibility, the new user is sometimes overwhelmed and discouraged from using these powerful tools. However, learning how to use the tools of SAS/GRAPH can be a pleasant experience.

This paper introduces several ways to generate graphical representations of your data. Starting with the importance of understanding their data, users are lead in a step by step process that ends with the generation of high quality plots and graphs. Finally, the paper ends with a short explanation on how to use the Annotate facility of SAS/Graph. Let us begin.

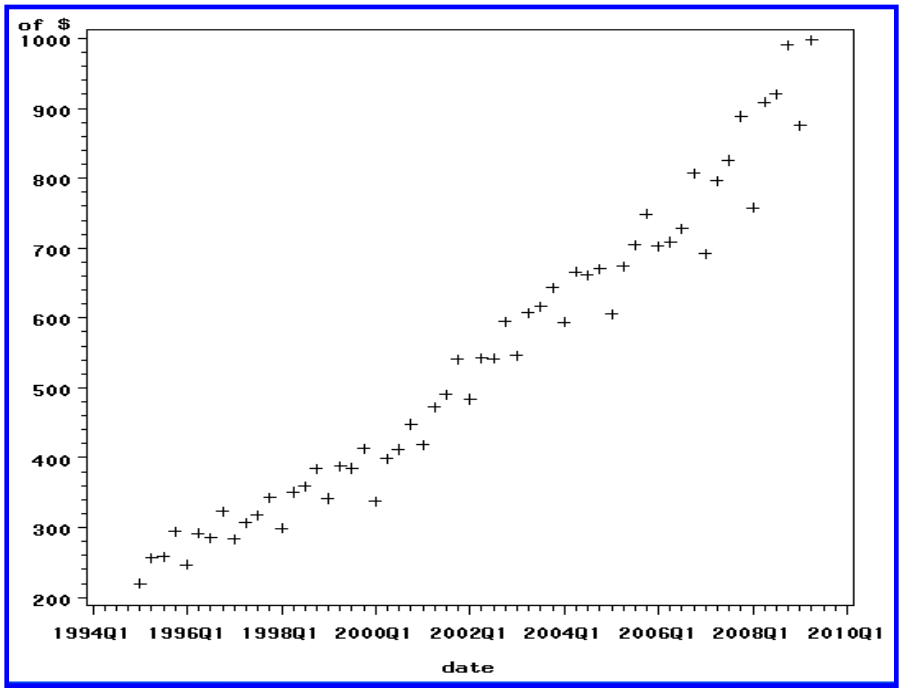
Step 1: Understanding Your Data. Is some data more appropriate for generating graphical reports than other data? What data is most appropriate for plots and graphs? The first data set to be examined is the SASUSER.RETAIL table.

	Retail sales in millions of $	date
1	220	1995Q1
2	257	1995Q2
3	258	1995Q3
4	295	1995Q4
5	247	1996Q1
6	292	1996Q2
7	286	1996Q3
8	323	1996Q4
9	284	1997Q1
10	307	1997Q2

A quick look at the first ten rows reveals only 2 columns or variables: SALES, and DATE. The SALES variable has a descriptive label associated with it. Both variables are numeric. One contains monetary values, while the other contains dates. They are both what we would call continuous numeric, in that their values could plotted on a number line. This data set is perfect for 'plotting'.
Step 2: Generating Plots From Your Data. With SAS/GRAPH you can use the GPLOT procedure to plot one variable against another. In this case, we are going to plot 'money' over 'time'. When you have a variable that represents some component of time, it is usually plotted on the horizontal axis, while the 'other' variable is plotted on the vertical axis. It is very easy to tell the SAS system to draw this kind of plot by using the following code:

```
proc gplot  data=sasuser.retail;
   plot sales * date;
run;
```

The first statement invokes the GPLOT procedure on the SASUSER.RETAIL data set. The first variable mentioned on the PLOT statement is plotted against the vertical axis, while the second variable is plotted against the horizontal axis. This simple code produces the following plot:



This plot is fairly descriptive and you can definitely see the trend throughout time. It's a good trend for this particular organization. However, the plot is somewhat plain. We can enhance the plot by using several of the options available through the PLOT procedure. Let's enhance this plot by: defining a plot symbol, joining the symbols by straight line segments, and selecting a color. To do this, we will need to use the SYMBOL statement.

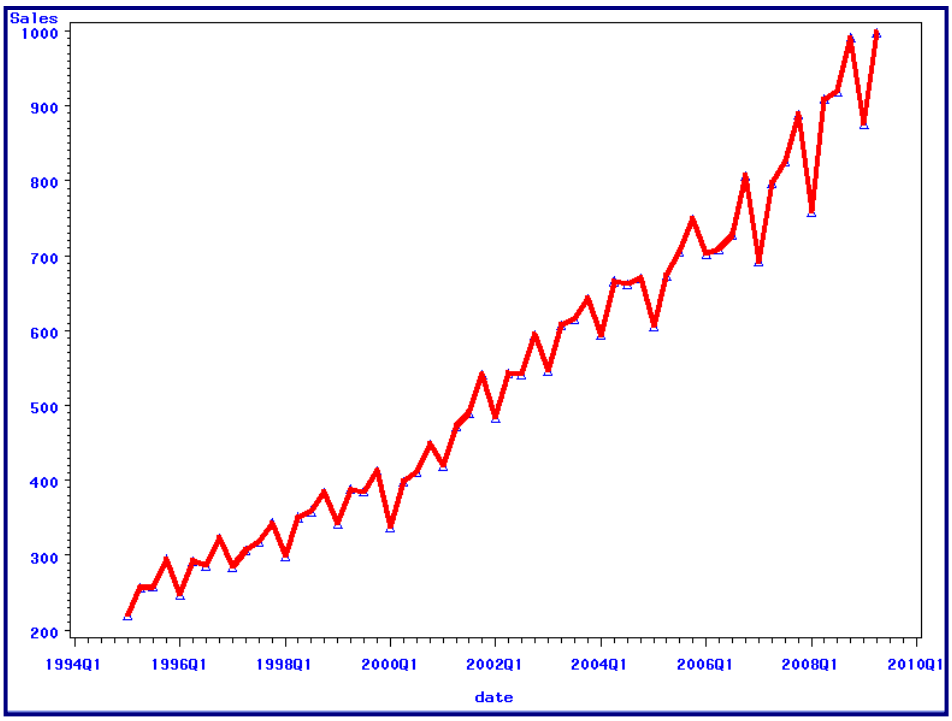
The SYMBOL statement allows you to specify the several options at one time that can enhance your plot. Selected options are:

- V or VALUE : specifies the plot symbol
- I : specifies the interpolation method
- CI : specifies the color of the interpolation method
- CV : specifies the color of the plotting symbol
- W or WIDTH : specifies the line thickness.

Next, lets enhance the plot by using the SYMBOL and LABEL statements.

```
proc gplot data = sasuser.retail ;
    plot  sales  *  date  ;
    symbol  v = triangle I = j ci=red  w = 5;
    label sales = 'Sales';
run;
```

The above SYMBOL statement draws a thick red line through triangles. The LABEL statement shortens the previously defined label to just 5 characters. The output looks like this:



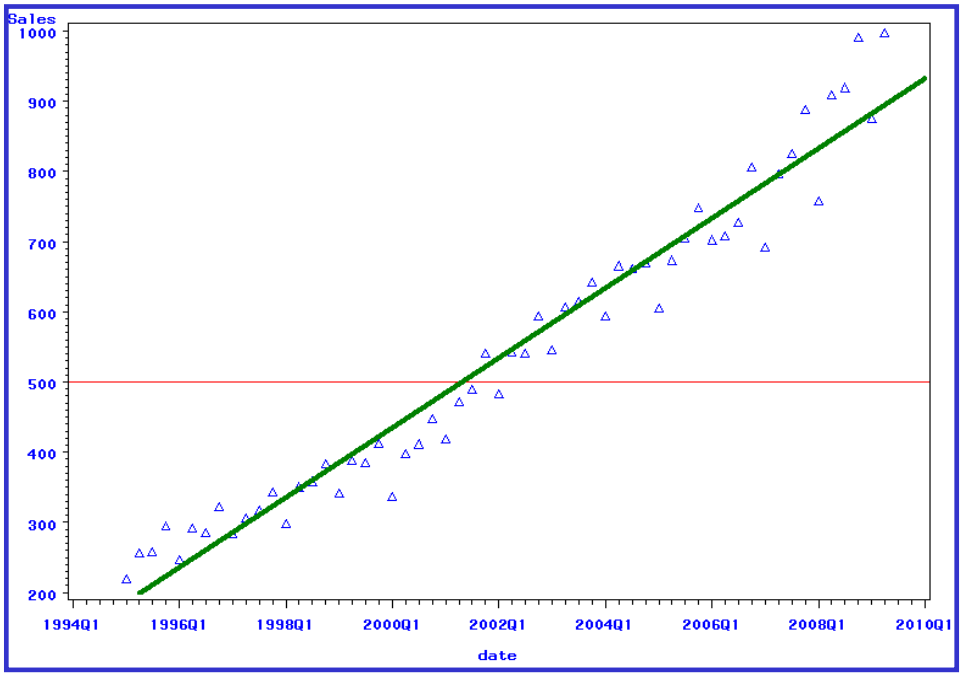
The next step is to enhance the plot by doing the following:

- drawing a reference line
- drawing a green linear regression line

To do this, we will modify the SYMBOL statement as well as use some PLOT statement options. Specifically, we will use the VREF option to draw reference line that references the VERTICAL axis where the value is 500. The LVREF option will be used to make the reference line a solid one. And the CVREF option will be used to make the reference line red.

```
proc gplot data = sasuser.retail ;
     plot  sales  *  date  / vref = 500
             cvref=red lvref=1;
     symbol  v = triangle  i = rl
             ci = green  w = 5 ;
     label sales = 'Sales';
 run; quit;
```

The results are shown below:



Other PLOT enhancements will be shown during the presentation of this paper.

Step 3: Generating Bar Charts. You can use SAS/GRAPH to generate fairly simple bar charts. We will do this initially, then go through a series of steps to enhance the output. To generate BAR CHARTS, you use the GCHART procedure.

With the GCHART procedure, you can not only generate BAR CHARTS, but you can also draw PIE and BLOCK charts as well. First, lets take a look at the data.

A modified version of the SASHELP.MDV data set will be used for the PROC GCHART examples. The first ten rows of the data are shown below:

Obs	CODE	COUNTRY	TYPE	CITY	sales2002	sales2001	sales2000	sales1999
1	NEXT DAY	AUSTRALIA	MD11	ACTON	1660.57	1500.24	523.24	288.24
2	NEXT DAY	AUSTRALIA	DC10	ACTON	874.62	804.24	523.24	499.24
3	SECOND DAY	AUSTRALIA	A300	ACTON	1340.82	1308.24	523.24	198.24
4	THIRD DAY	AUSTRALIA	MD11	MELBOURNE	1876.61	1596.00	1170.00	529.00
5	SECOND DAY	AUSTRALIA	DC10	MELBOURNE	1320.58	1178.29	251.29	628.71
6	NEXT DAY	AUSTRALIA	747	MELBOURNE	1237.36	1144.29	251.29	149.71
7	SECOND DAY	AUSTRALIA	A300	MELBOURNE	1318.61	1106.29	251.29	640.71
8	NEXT DAY	AUSTRALIA	777	MELBOURNE	1077.62	954.29	251.29	418.71
9	NEXT WEEK	CANADA	MD11	MONTREAL	309.77	268.00	1120.00	948.00
10	SECOND DAY	CANADA	DC10	MONTREAL	1738.20	1522.00	834.00	121.00

When using the GCHART procedure, you need to specify three things. First, you specify the physical form of the chart. Next, you need to identify the CHART variable as well as the ANALYSIS variable.

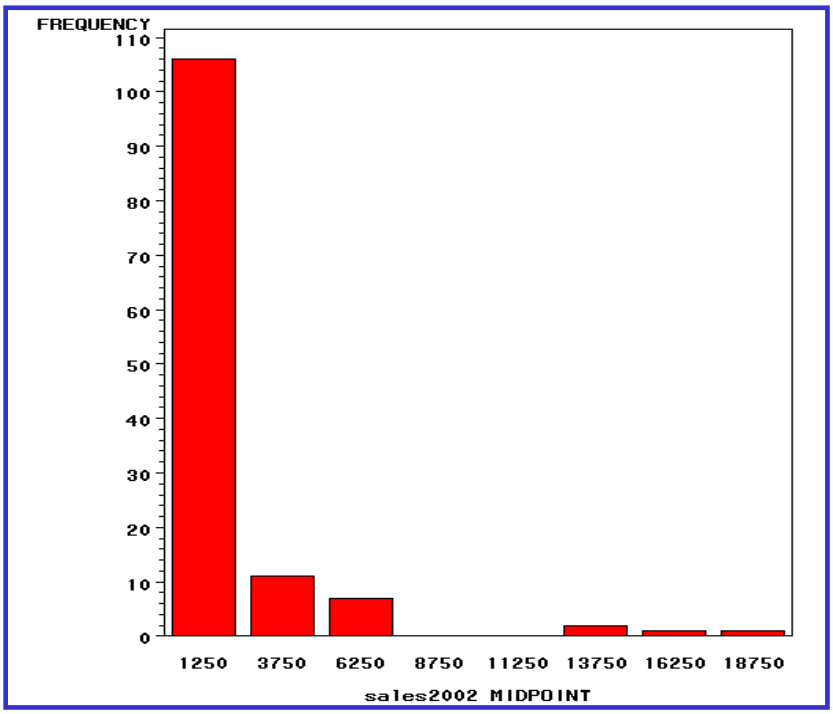
The CHART variable determines how many BARS or SLICES (for PIE charts). Usually, you will get one bar (or slice) for each unique value of the CHART variable. CHART Variables are usually character and have values that are discrete and categorical.

The ANALYSIS variable determines the height (or length) of the bars, or the size of the slice. These variables are used for calculating statistics and are always numeric.

To begin with, lets draw a simple BAR CHART. The VP of sales wants to know how much money was made in 2002. So, as a first attempt to find out, the following program was written:

```
proc gchart data = sasuser.mdv2002;
    vbar sales2002;
run;
```

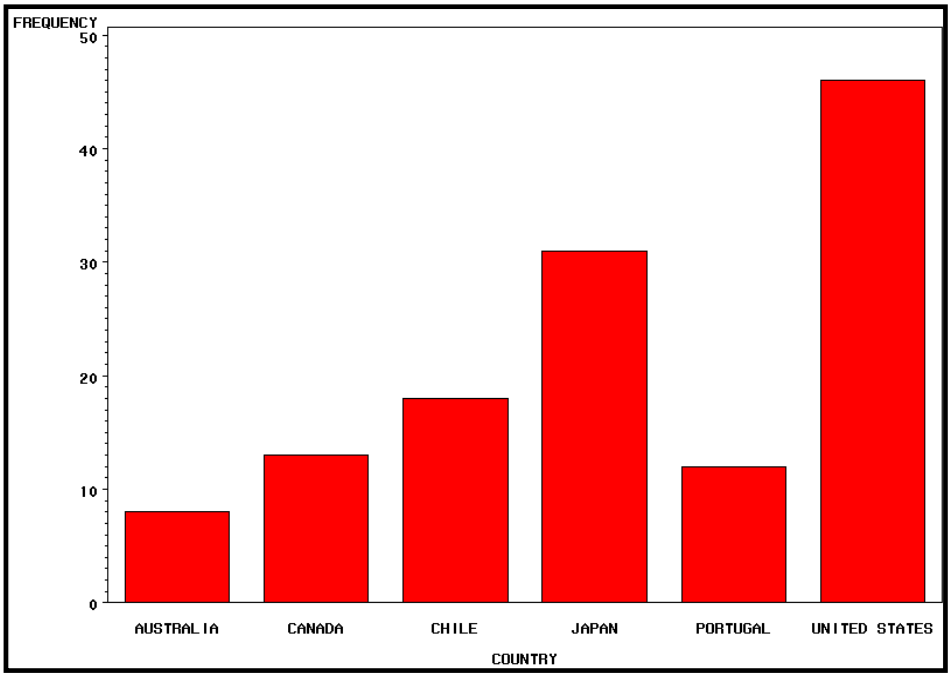
First, a few things need to be pointed out. The VBAR statement determines the physical form of the chart. A vertical bar chart will be generated from this code. Next, SALES2002 is the CHART variable. Does it have discrete and categorical values? What will determine the height of each bar? Lets look at the output.



Is this what we expected to see? In this case, a numeric charting variable was used. If bars were drawn from each unique value, there would be a lot of bars. What the GCHART procedure did was to automatically calculate the intervals and identify them as MIDPOINTS for SALES2002.
This is not usually what we expect to see. So, lets try this again with a different CHART variable.

```
proc gchart data = sasuser.mdv2002;
    vbar country;
run;
```

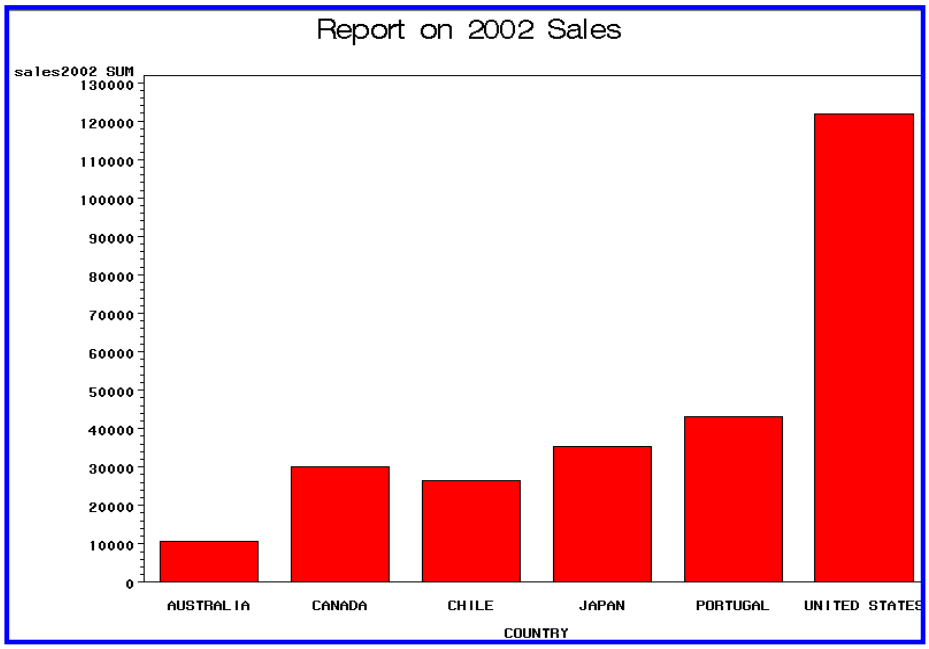
The variable COUNTRY does have values that are discrete and categorical. What will determine the height of each bar? Lets look at the output.



As you can tell from the output, the height of the bars is determined by a FREQUENCY count. In other words, there were 8 observations in this data set that had AUSTRALIA as the value for COUNTRY, there were 12 observations in this data set that had CANADA as its value for COUNTRY. So, FREQUENCY is the default statistic. Redraw the graph, only this time, let the height of the bar reflect the total sales for a specific year.

```
proc gchart data = sasuser.mdv2002;
   vbar country / sumvar = sales2002;
   title 'Report on 2002 Sales';
run;
```

The **SUMVAR= option** tells the GCHART procedure that we want the height of each bar to be the SUM of SALES2002. Also, we are adding a title to the output.



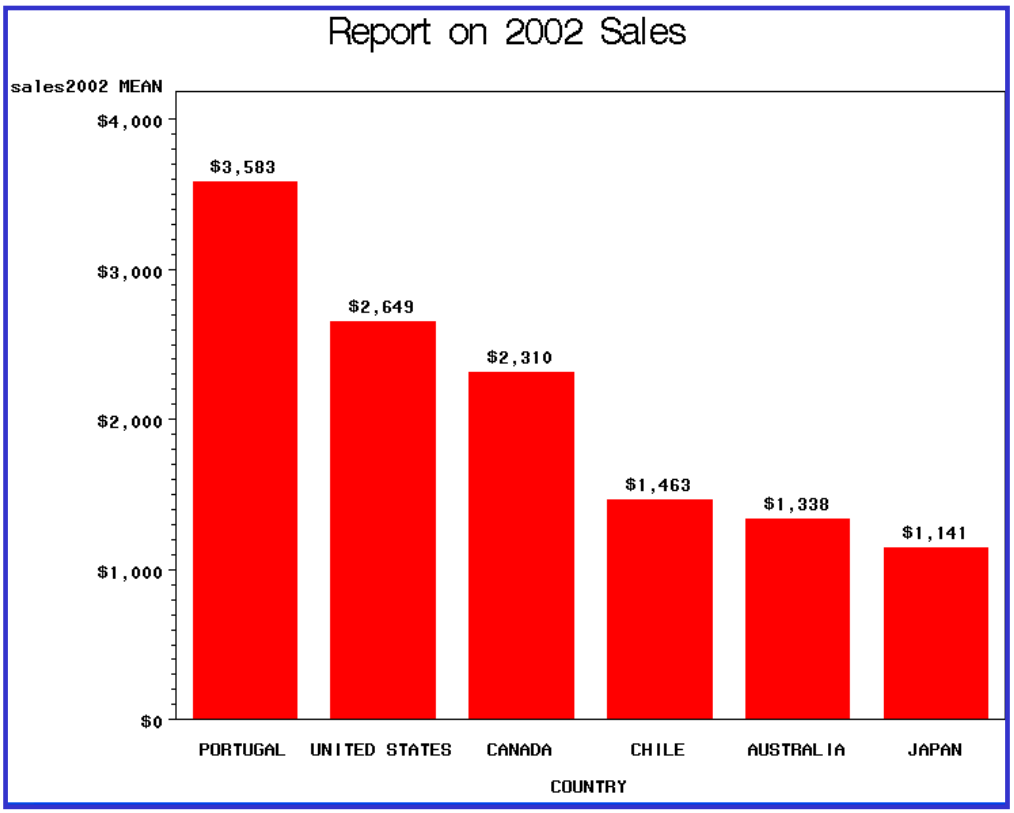
Lets enhance the output further by doing the following:

- Order the bars by using a descending option on the VBAR statement.
- Make the bars represent the AVERAGE sales (instead of TOTAL sales)
- Place the AVERAGE sales value on the 'top' of each bar.

Use the following code to accomplish these enhancements.

```
proc gchart data = sasuser.mdv2002;
   vbar country / sumvar = sales2002
                  type = mean  mean
                  descending;
   title 'Report on 2002 Sales';
run;
```

The **TYPE=** option specifies the MEAN statistic. The second 'MEAN' indicates you want the MEAN statistic to be displayed above each bar. The DESCENDING keyword tells the GCHART procedure to display the bars in descending height. The output is shown below:



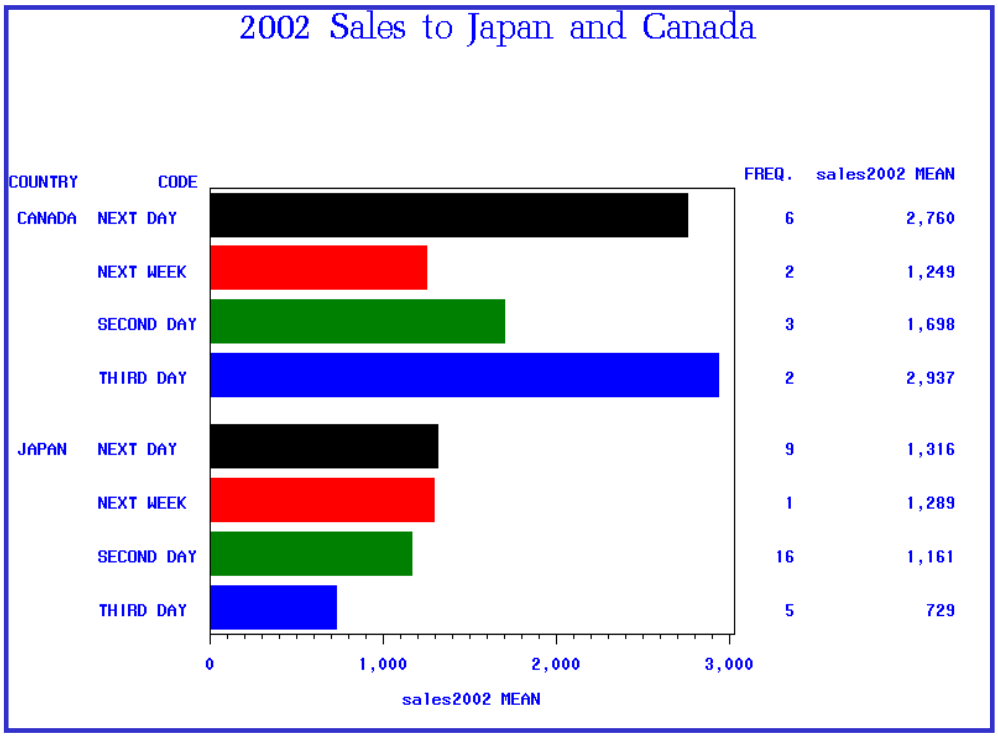
It's easy to change the physical form of the chart. Change the VBAR statement to HBAR. Since we do not want to draw a chart on the entire data, we will use a WHERE statement to subset the data. Other enhancements will be explained after we examine the code.

.

```
proc gchart data = sasuser.mdv2002;
   hbar code /  type = mean
                group = country
                sumvar = sales2002;
                patternid = midpoint;
   title c=blue f=zapf   '2002 Sales to Japan and Canada';
   where country in('JAPAN', 'CANADA');
run;
```

You will notice the use of the following options:

- GROUP= to group bars,
- PATTERNID=MIDPOINT to give each bar a different color,
- TITLE statement options to control the color and font of the title text.



You can also store your graphic output as a GIF or JPEG file rather than in a SAS catalog by using the following code:

```
filename out 'c:\test.gif' ;
goptions dev = gif  fsfname = out;

proc gchart data=sasuser.mdv2002;
     hbar code /  sumvar = sales2002
                  type = sum
                  group = country
                 subgroup = type;
      title  c=red '2002 Sales to Canada and Japan' ;
     where country in('AUSTRALIA','JAPAN');
     format sales2002 comma12.;
run;   quit;
```

Notice the first two statements.

The notes in the SAS/Log indicate that the barchart was indeed written to a GIF file.

```
470  filename out 'c:\test.gif';
471  goptions dev=gif gsfname=out;
472  proc gchart data=sasuser.mdv2002;
473     hbar code / sumvar = sales2002
474           type = sum
475           group= country
476           subgroup = type;
477     title c=red '2002 Sales to Canada and Japan';
478     where country in('AUSTRALIA', 'JAPAN');
479     format sales2002 comma12.;
480  run;

NOTE:  65 RECORDS WRITTEN TO c:\test.gif
480!       quit;
```

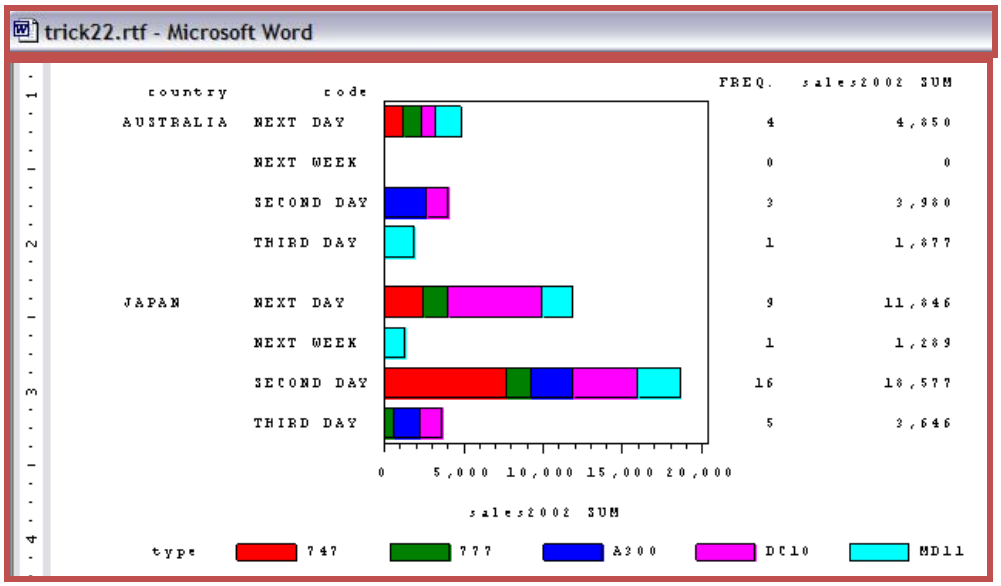
The ODS (Output Delivery System) can be used to write the barchart to a document (rtf file) that can be read by Microsoft Word. This can be done by using two ODS statements and wrapping them around the previous GCHART step.

```
ods rtf file='c:\trick22.rtf';

proc gchart data=sasuser.mdv2002;
     hbar code / sumvar=sales2002
     group=country
     subgroup=type;
     where country in('AUSTRALIA', 'JAPAN');
     format sales2002 comma12.;
run;

ods rtf close;
```

The output is sent to a rtf file that can be read by Microsoft Word.



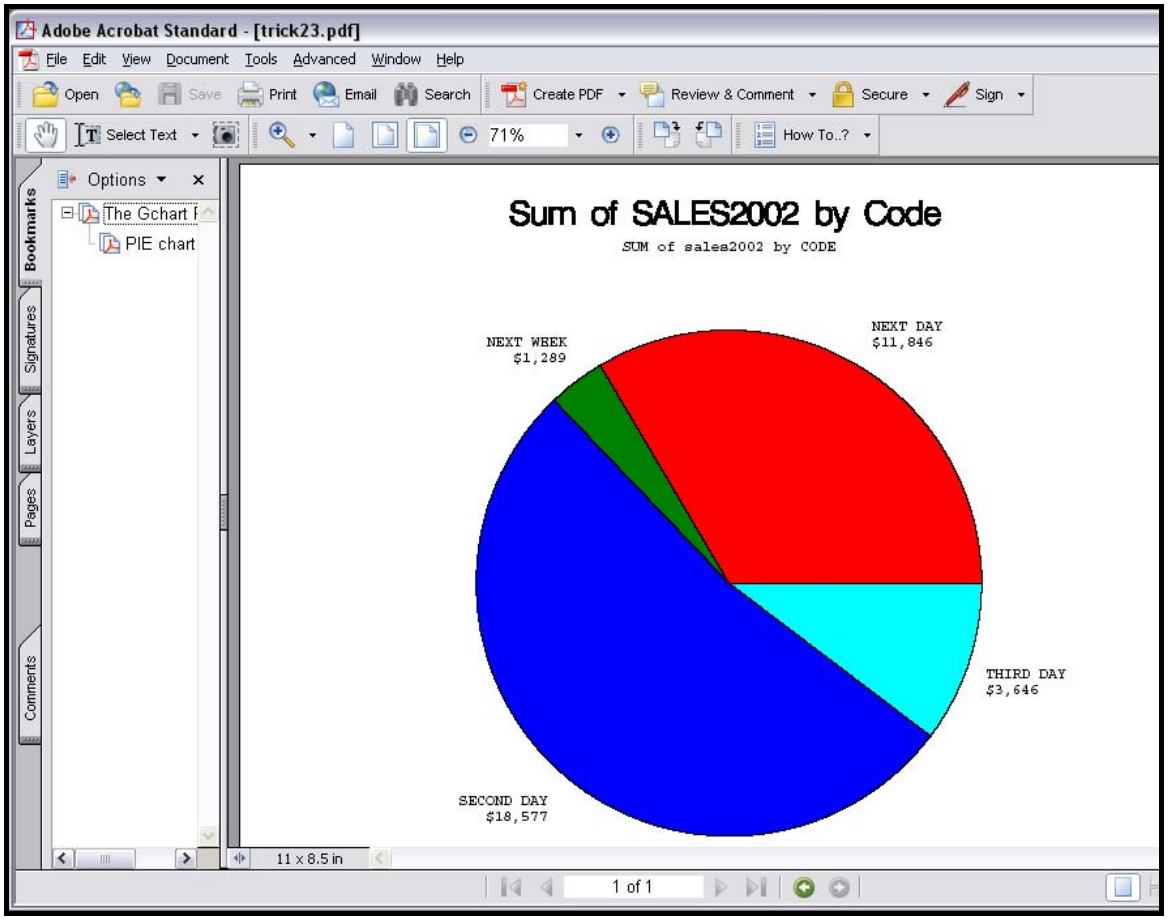
Next, use the GCHART procedure to generate a pie chart where we will have a slice for each value of the CODE variable. At the same time, modify the ODS statements to send the output to a PDF file.

```
ods pdf file='c:\trick23.pdf';

title 'Sum of SALES2002 by Code';
proc gchart data=sasuser.mdv2002;
        pie code / sumvar=sales2002;
        where country in('AUSTRALIA', 'JAPAN');
        format sales2002 dollar12.;
run; quit;

ods pdf close;
```

Using Adobe Acrobat to read the pdf file, we can view the report



SAS/Annotate Facility

The Annotate Facility of SAS/Graph allows users to customize their graphic output. These customizations can be either predetermined or data-driven. This section will focus on the data driven modifications that can be made to your SAS/Graph output. The Annotate Facility is a part of the SAS/Graph product. Its power is

accessed through the use of a specialized dataset known as the Annotate Data Table. This table is a regular SAS data set that contains specific variables that answer the following questions**:**

- What is to be done?
- How is it to be done?
- Where is it to be done?

For example, an annotate data set can allow us to put stars on a map exactly where certain cities are located. One data set is used to draw the map (maps.us), and another one (work.citystar1) is used to annotate the map.

One of the two Annotate datasets used in this section is shown here.

VIEWTABLE: Work.Citystar1

	function	style	color	position	text	xsys	ysys	hsys	when	X	Y	STATE	CITY	size
1	symbol	marker	red		V	2	2	3	a	0.1196016127	-0.048502021	01	Kansas	4
2	label		green	8	Kansas	2	2	3	a	0.1196016127	-0.048502021	01	Kansas	7
3	symbol	marker	red		V	2	2	3	a	0.1059315226	0.0498526833	17	Kansas	4
4	label		green	8	Kansas	2	2	3	a	0.1059315226	0.0498526833	17	Kansas	7
5	symbol	marker	red		V	2	2	3	a	0.014896602	-0.013266246	40	Kansas	4
6	label		green	8	Kansas	2	2	3	a	0.014896602	-0.013266246	40	Kansas	7

Questions these variables answer:

- **What** - FUNCTION
- **How** - COLOR, SIZE, STYLE, POSITION
- **Where** – X, Y, XSYS, YSYS

The variable FUNCTION is the key to the annotate process. Once the FUNCTION has been determined (WHAT), its supporting variables are selected (HOW) and then finally the location variables (WHERE) are determined. In this example, there are only two unique values of FUNCTION: symbol and label. Symbol observations create the red stars, while the label observations create the green labels of 'Kansas'.

Step 1: Write a DATA Step to create the Annotate data set.

```
 /* create the Annotate data set, CITYSTAR1 */
data citystar1;
   length function style color $ 8 position $ 1 text $ 20;
   retain xsys ysys '2'  hsys '3'  when 'a';
     set maps.uscity(keep=x y city state where=(city in('Kansas')));
     function='symbol'; style='marker'; text='V'; color='red'; size=4;
     output;
     function='label'; style=''; text=city; color='green'; size=7; position='8';
     output;
 run;
```

Step 2: Create a subset of states (unless you want to use ALL 48 of the states).

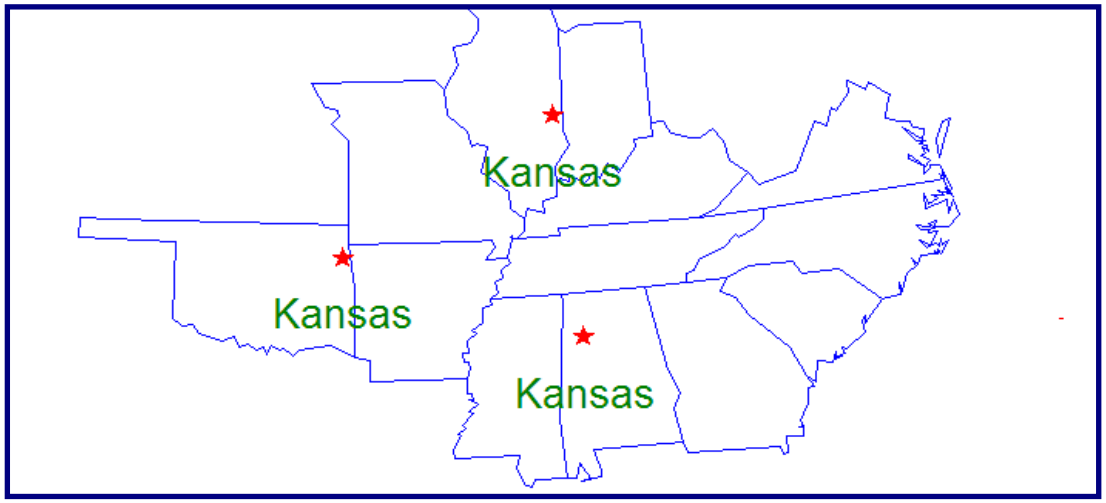
```
data central_states;
      set maps.us;
      if statecode in('GA' 'AL' 'MS' 'IN' 'AR' 'IL' 'MO' 'KY' 'TN'
                      'OK' 'NC' 'SC' 'VA');
 run;
```

Step 3: Generate the map with PROC GMAP and ANNOTATE it with the Annotate dataset.

```
pattern value=mempty color=blue;

proc gmap data= central_states  map=central_states;
   id state;
   choro state / annotate=citystar1 discrete
                 nolegend;
run; quit;
```

The **central_states** data set draws the map, while the **citystar1** dataset 'annotates' the map with green labels and red stars as shown below.



Other enhancements will be shown when this paper is presented.

CONCLUSION

There are a number of ways to generate graphic output with the SAS system. Methods other than SAS/GRAPH may be easier, but none has near the power and control found in the SAS/GRAPH syntax. With its dozens of options and statements, a user can customize their graphic output to create exactly what they want. For more information on SAS/GRAPH, see the documentation offered by SAS Institute.
Go to WWW.SAS.COM and select Services, then Publications.

The author can be reached at:

Ben Cochran
The Bedford Group
(919) 741-0370
bedfordgroup@nc.rr.com

SAS Alliance
Affiliate Member

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