SESUG Paper 255-2019

Easier and Better with SAS® Enterprise Guide

Laura Oliver, Experis Solutions

ABSTRACT

SAS® Enterprise Guide (EG) is a great Integrated Development Environment that helps developers do the full advanced analytics lifecycle through a point-and-click or hard-core programming interface. Many times, though, the user hasn't had time to investigate all of the features in EG and ways to make life easier. For example, the GUI environment has good ways to accomplish tasks that had to be done in a program in the past.

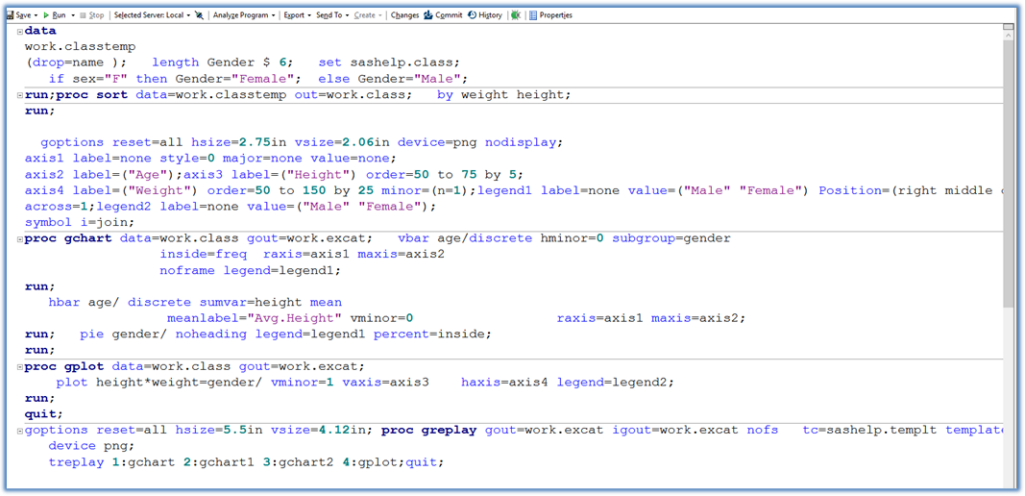
INTRODUCTION

This paper is a companion to the Hands-On-Workshop (HOW), Easier and Better with SAS® Enterprise Guide. . It is designed to show some of the shortcuts that Enterprise Guide has to help make tasks easier. They can be useful for developers as well as statisticians who prefer to point and click and get results and everyone in between. The Data Step Debugging feature as well as the Query Builder will be investigated along with how to get the most out of tasks. Enterprise Guide 7.13 will be used throughout the examples.

COOL FEATURES IN ENTERPRISE GUIDE

FORMATTING CODE

Many times developers are asked to review and modify code developed by other individuals. These individuals may have a different style or different requirements to the formatting of the code. Or there may not be any formatting at all. Enterprise Guide has created a way to assist when this happens. In the code below the procedures are not formatted in an easy to read manner. This makes it hard to follow what is going on and makes it harder to maintain. With just a few clicks with the mouse, the code can be formatted to make it readable.



```
data
 work.classtemp
 (drop=name );   length Gender $ 6;   set sashelp.class;
    if sex="F" then Gender="Female";  else Gender="Male";
run;proc sort data=work.classtemp out=work.class;   by weight height;
 run;

   goptions reset=all hsize=2.75in vsize=2.06in device=png nodisplay;
 axis1 label=none style=0 major=none value=none;
 axis2 label=("Age");axis3 label=("Height") order=50 to 75 by 5;
 axis4 label=("Weight") order=50 to 150 by 25 minor=(n=1);legend1 label=none value=("Male" "Female") Position=(right middle
 across=1;legend2 label=none value=("Male" "Female");
 symbol i=join;
proc gchart data=work.class gout=work.excat;   vbar age/discrete hminor=0 subgroup=gender
                inside=freq  raxis=axis1 maxis=axis2
                noframe legend=legend1;
 run;
    hbar age/ discrete sumvar=height mean
                  meanlabel="Avg.Height" vminor=0                  raxis=axis1 maxis=axis2;
 run;   pie gender/ noheading legend=legend1 percent=inside;
 run;
proc gplot data=work.class gout=work.excat;
     plot height*weight=gender/ vminor=1 vaxis=axis3    haxis=axis4 legend=legend2;
 run;
 quit;
goptions reset=all hsize=5.5in vsize=4.12in; proc greplay gout=work.excat igout=work.excat nofs   tc=sashelp.templt templat
    device png;
    treplay 1:gchart 2:gchart1 3:gchart2 4:gplot;quit;
```

Figure 1 - Before Formatting

The following steps will accomplish the formatting:

1. In an open program if no code is selected, formatting will be applied to all code in the program. If only

a portion of the code needs to be formatted, that portion can be selected and then formatted

2. Select Edit → Format Code or Ctrl I and the code is formatted.

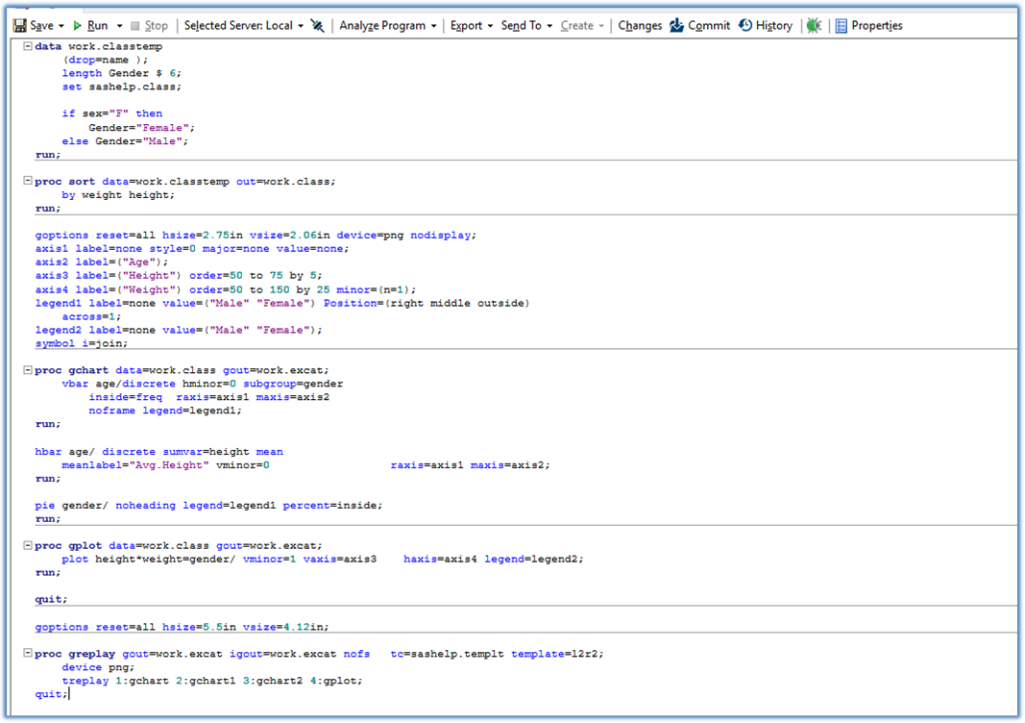


Figure 2 - After Formatting

After formatting code the first time, the spacing may not match requirements or preferences. The formatting may have put an extra line after the set statement. Or the formatting of then/else/do does not meet requirements. Under the Tools menu, some of the formatting options can be changed.

Tools → Options, click on SAS Programs, then Editor Options, Indenter Tab.

Or

Program → Editor Options, Indenter Tab

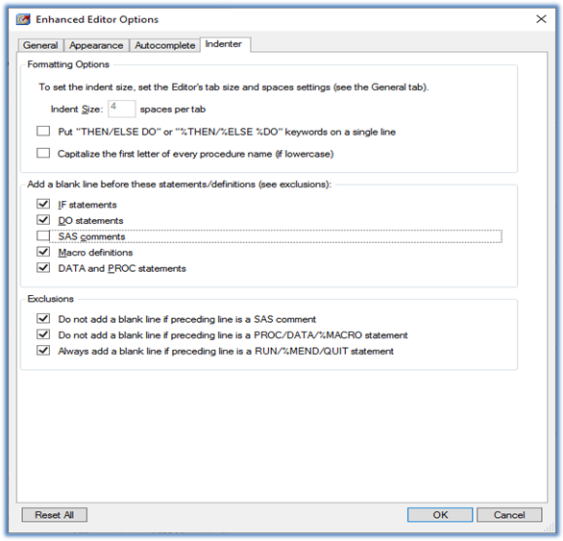


Figure 3 - Enhanced Editor Options

LAYOUT ARRANGEMENT

When troubleshooting code, there can be a need to compare items. EG allows the opening of multiple windows similar to how it was done in SAS Display Manager. The workspace layout can be changed under the View menu. The drop down arrow for each window is used to open the code/dataset/log to be viewed.

View → Workspace Layout → Stacked

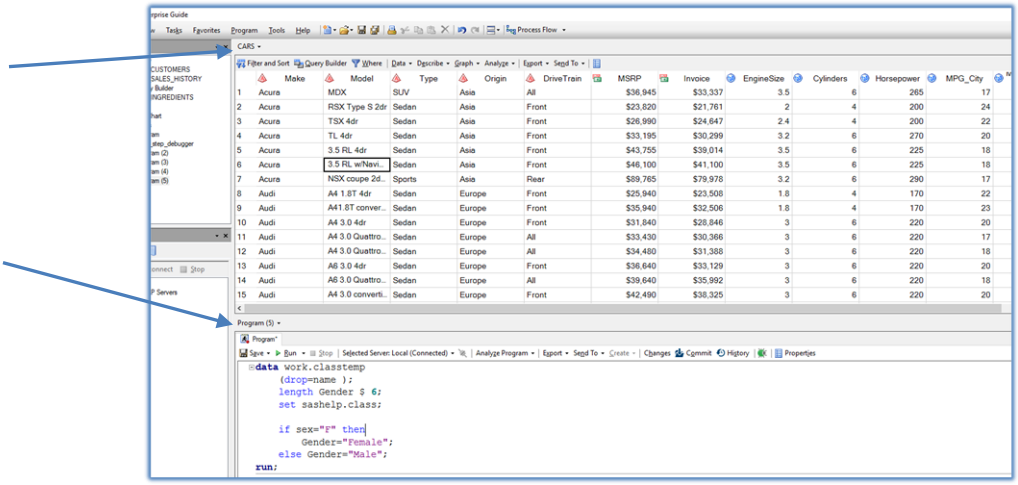


Figure 4 - Stacked Workspace Layout

FAVORITES

EG has a Favorites menu just like a web browser. Once a program and/or data has been added to a project, it can be added to Favorites. The program/data does not have to stay in the project, it just has to be in the project to be added to Favorites. EG Favorites are managed the same as browser Favorites. A folder can be created to organize favorites, the order of favorites can be changed and favorites can be deleted when no longer needed. But a folder cannot be added to Favorites, only data and programs.

SAS MACRO VARIABLE VIEWER

One of the key elements of SAS is macro variables. These can be created in code, or automatically created by the system. The %put _all_; code can be used to list all macros variables and their values. But another way to view them is using the SAS Macro Variable Viewer. This will display all macro variables and their values. It can also be used to evaluate a macro. The **SAS Macro Variable Viewer** is under the Tools menu

Tools –> SAS Macro Variable Viewer

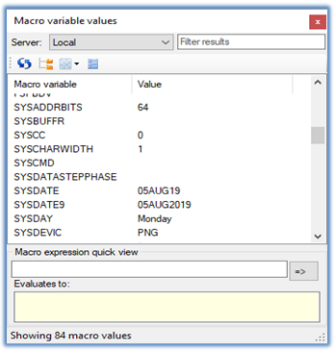


Figure 5 - Macro Variable Viewer

If the following code is run in EG and then the **SAS Macro Variable Viewer** is opened again, the new macro is now displayed with its value in the list.

```
%let myday = Monday;
```

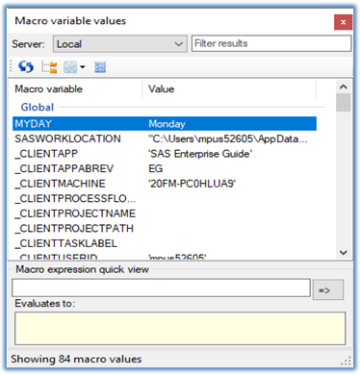


Figure 6 - Macro Variable Viewer After Code

SYSTEM OPTIONS VIEWER

Along with the ability to view macro variables using a menu, the system options can be viewed the same way. The **Systems Options Viewer** is under the Tools menu

Tools → System Options Viewer

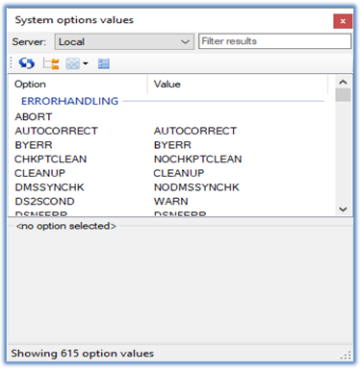


Figure 7 - System Options Values

DATA STEP DEBUGGER

When coding, there are times when the code is not doing what is expected. It has passed the syntax check done by SAS at compile time, but the results are not right. It would be helpful to debug a data step line by line. This can be done using the ldebug option after the data statement, or by using the PUT statement to show what values are changing at each step. But EG has created a user friendly way to accomplish this task. On the menu bar starting in 7.13, there is now a green bug which acts as a toggle to turn the debugger on and off.

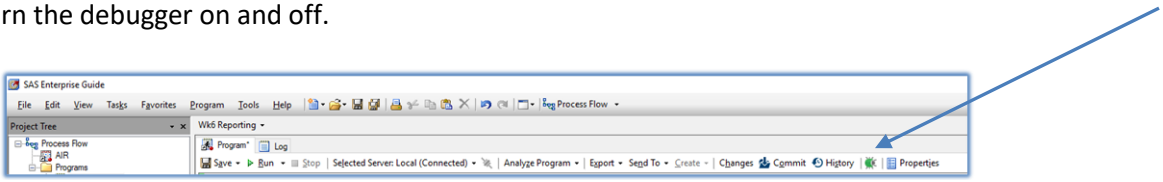


Figure 8 - Data Step Debugger Icon

When the Data Step Debugger is on there is a green highlighted area to the left of the data step being debugged. If this area is clicked, the **Debugger Window** is opened and ready use. And just as it indicates, it is a data step debugger, not a procedure debugger, and it only does 1 data step at a time.

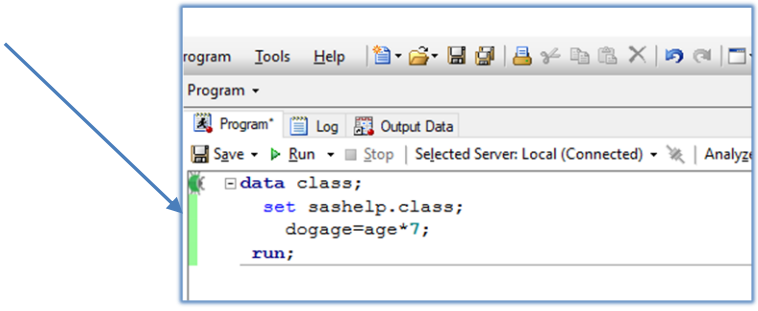


Figure 9 - Data Step Debugger Turned On

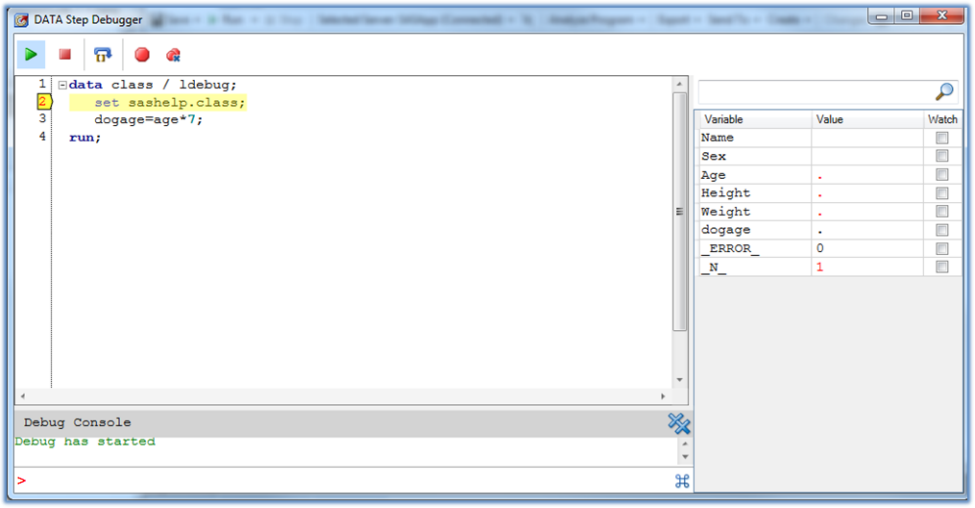


Figure 10 - Data Step Debugger Window

Using the following code, the data step debugger can be used in a very simplistic manner to get a feel for how it works.

```
data class;
  set sashelp.class;
    if name = 'Henry' then
       name = ' ';
    length fstinitial $1;
    if name ^= ' ' then
       fstinitial = substr(name,1,1);
run;
```

1. Place the cursor anywhere within the data step and click on the green debugger
2. Click in the green highlighted area that appears on the left of the selected data step to open the Data Step Window.
3. Click on the third button from left to step through the program and see how the variable values change as the data step is stepped through.
4. Keep stepping through the program until it gets to Henry and observe how the results differ in the debugger window.

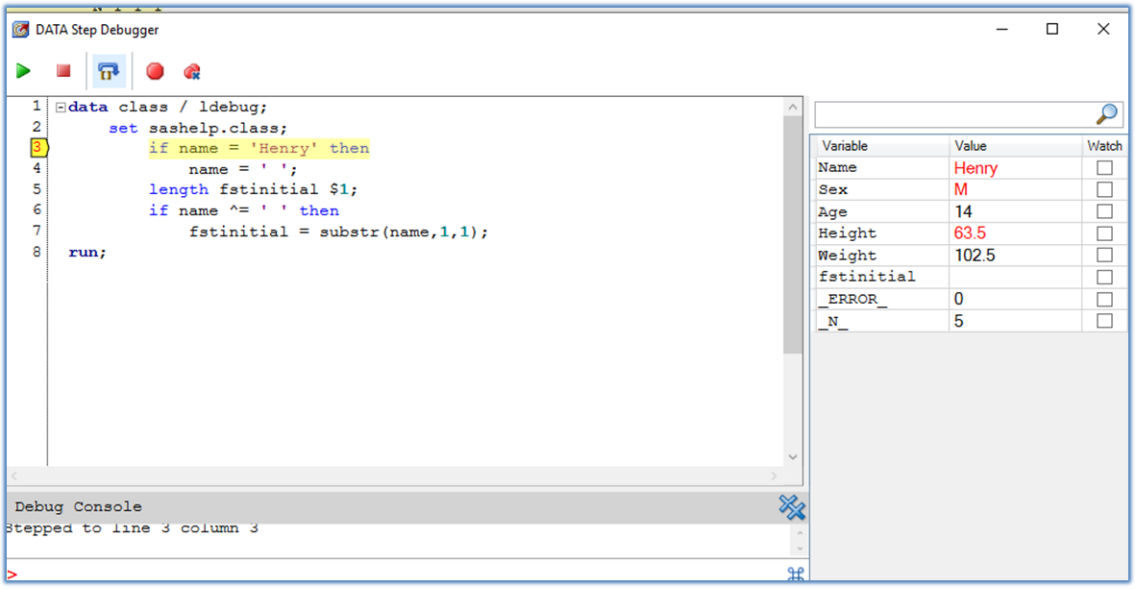


Figure 11 - Debugger Window

5. Click on the second button from left to terminate the debug process.
6. Close the debugger window.
7. Click the debugger button on the Program tab menu and turn the data step debugger off.

QUERY BUILDER BASICS

Enterprise Guide has a feature that allows users to query data without needing to write code. Experienced programmers may find that for most queries, it is easier to just write the code. But when there is a situation where multiple joins are needed, it might be helpful to use the query builder within EG. This way, the Query Builder's point and click functionality can be used to create the background code for the join, and then the code can be copied to use in a different program.

The candy_sales_history, candy_customers, and candy_ingredients tables are used in this query. A report needs to be created with all customers and the total units sold for candy that contains xylsugar.

As with most things in EG, Query Builder can be opened in multiple ways. It can be opened using the Data menu under the Tasks menu, opening a data set and then using the Query Builder button, or under the data menu in an open data set.

Let's create a query using the following steps:

1. Open the Candy_Ingredients data set.

2. Click the Query Builder menu item

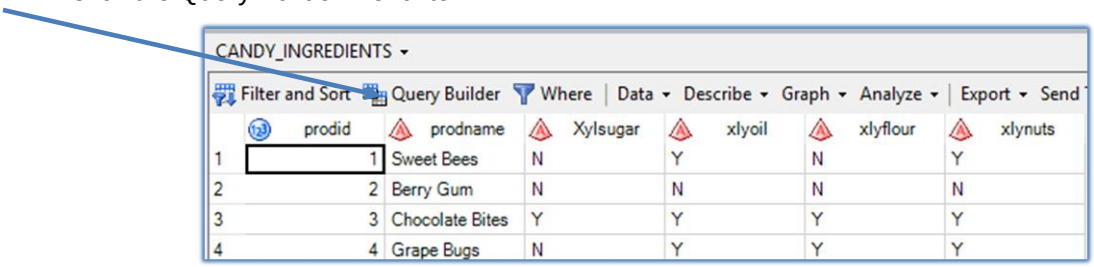


Figure 12 - Query Builder menu item

3. The Query Builder for Local:WORK.CANDY_INGREDIENTS dialog box will open.

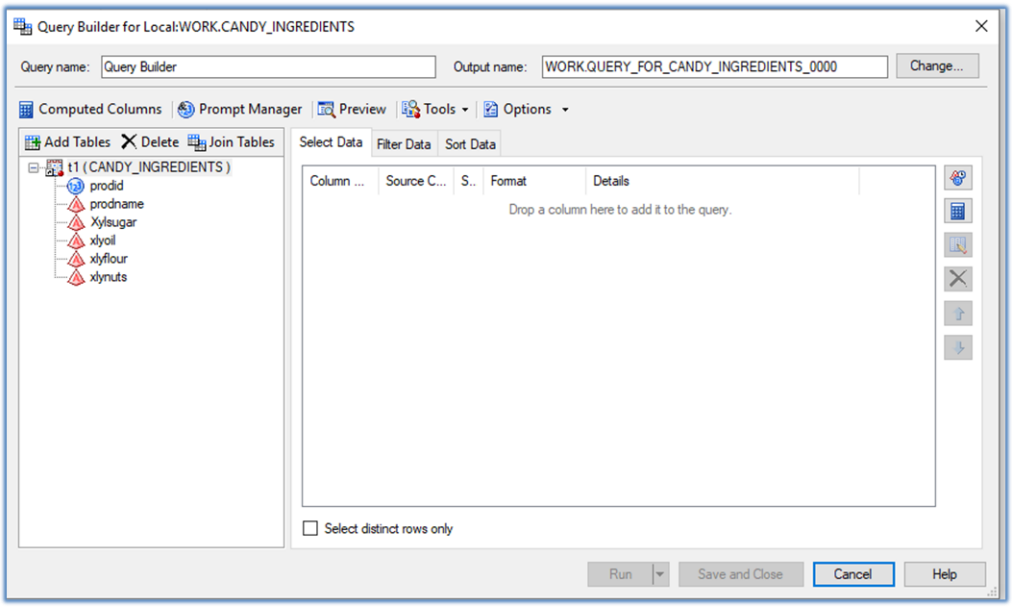


Figure 13 - Query Builder Window

4. Click the Add Tables Button to add the Candy_Customers and Candy_Sales_History data sets.

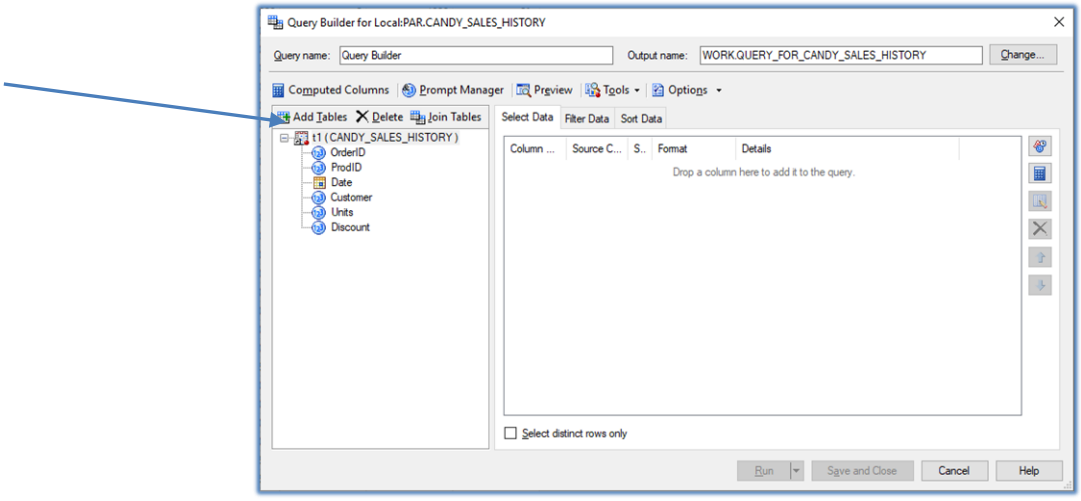


Figure 14 - Add Tables Button

5. Notice that the Candy_Sales_History data set was added without incident, but the Candy_Customers was not. EG could not decide what the join should be for those tables.

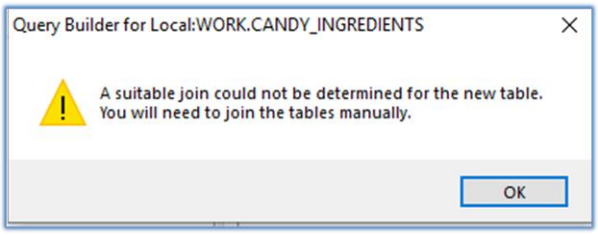


Figure 15 - Dialog Box for No Default Join

6. EG, joined Candy_Sales_History and Candy_Ingredients on prodid because the prodid field was in common. The Candy_Customers does not have a field in common so EG doesn't have a default join.
7. Click OK and use the **Tables and Joins** dialog box to join Candy_Sales_History to Candy_Customers on customer=custid by clicking on Customer and dragging to the custid field.

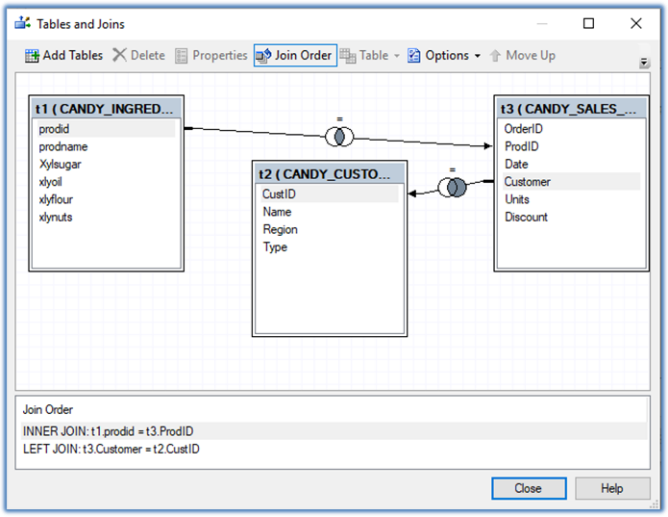


Figure 16 - Tables and Joins

8. The **Join Properties** box will be displayed after the tables have been visually joined. Select the Left Join in the Join Type list box and click OK.

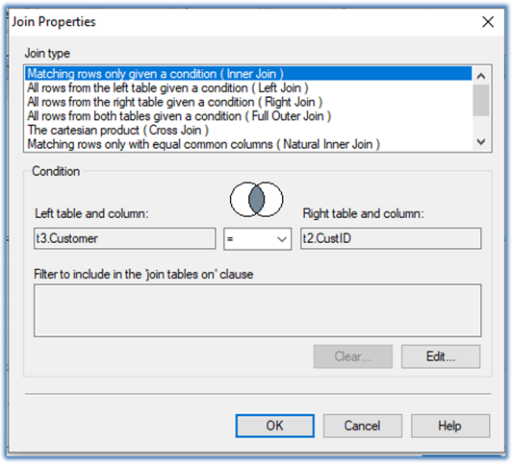


Figure 17 - Join Properties

9. Notice that at the bottom, the joins that are happening are shown. The join type can be changed by right clicking on the join in the diagram and clicking properties.
10. Close the **Join Tables** dialog box.
11. Click on the Select Data tab and add Name from Candy_Customers, prodname from Candy_Ingredients, and units from Candy_Sales_History to the query by double-clicking on the fields or by clicking and dragging to the Column in the Select Data tab area.

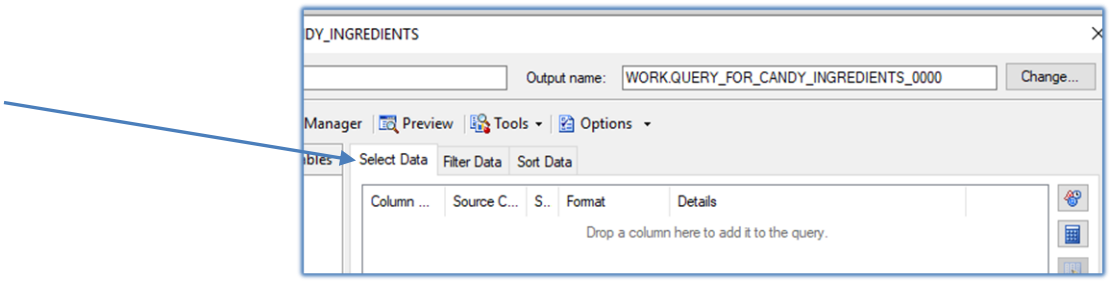


Figure 18 - Select Data tab

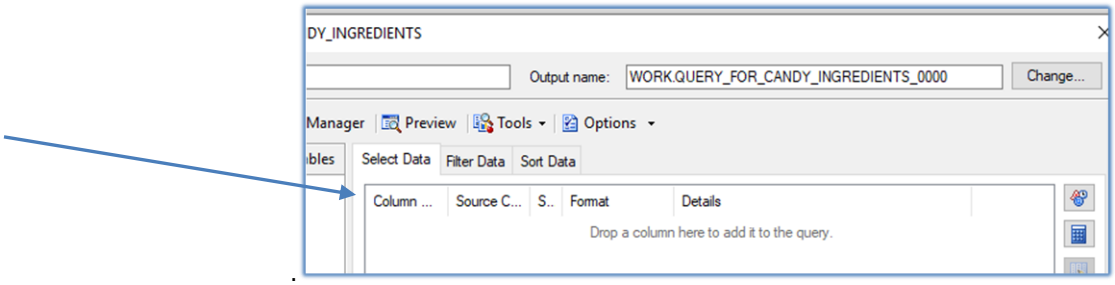


Figure 19 - Add Fields to Query

12. Click the drop down in the Summary column of the Units field and choose Sum

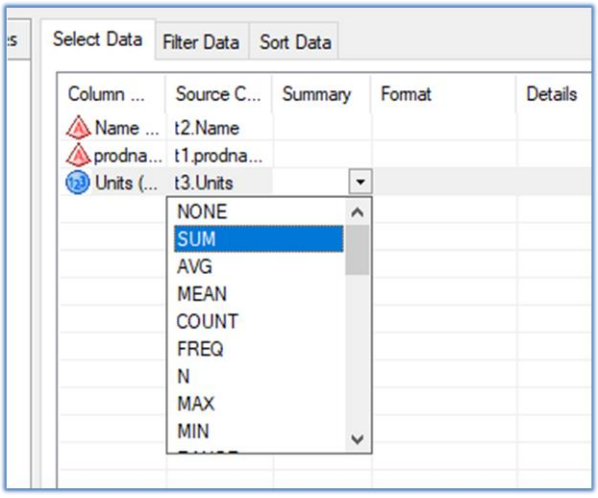


Figure 20 - Sum option in Summary Column

13. Double click in the Format column of the Units field and the Properties for Sum_Of_Units will open.
14. Click the Change button to apply Commaw.d format with an overall width of 9.
15. Click Ok
16. Click in the Column Name text box and change Total Units Sold

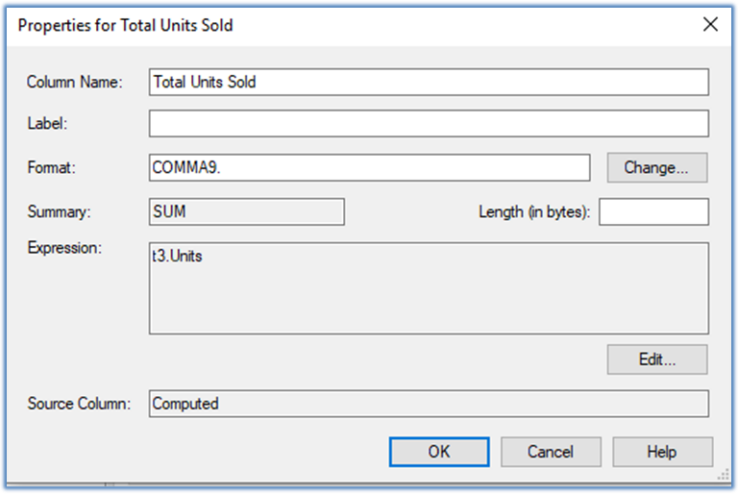


Figure 21 - Field Properties

17. Click OK
18. Click the Filter Tab
19. Click the New Filter Icon. 
20. Choose a Basic Filter on 1 of 4 and xlysugar from the candy_ingredients data set on 2 of 4.

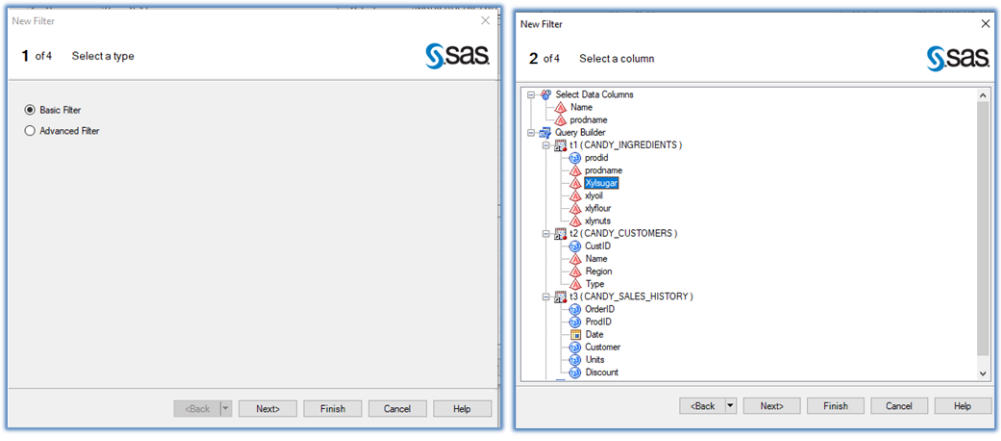


Figure 22 - New Filter Panels 1 & 2

21. Enter Y in the Value field on 3 of 4 and review the code on 4 of 4 and click Finish.

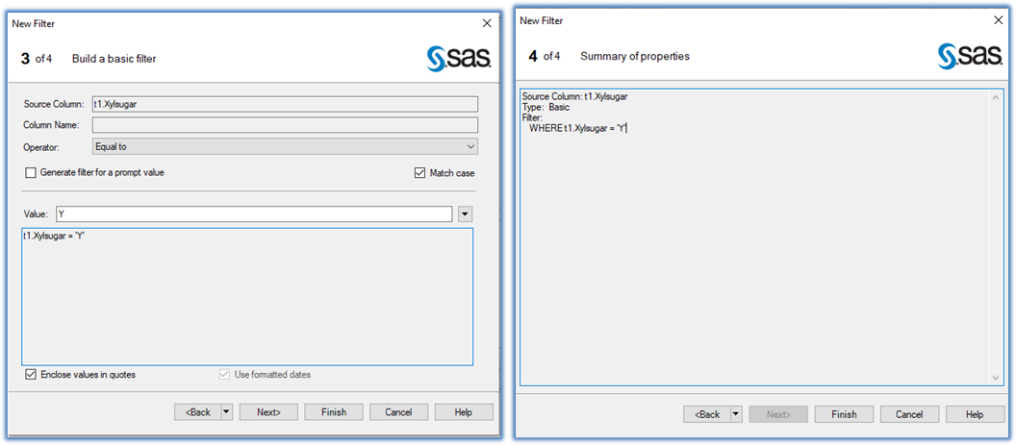
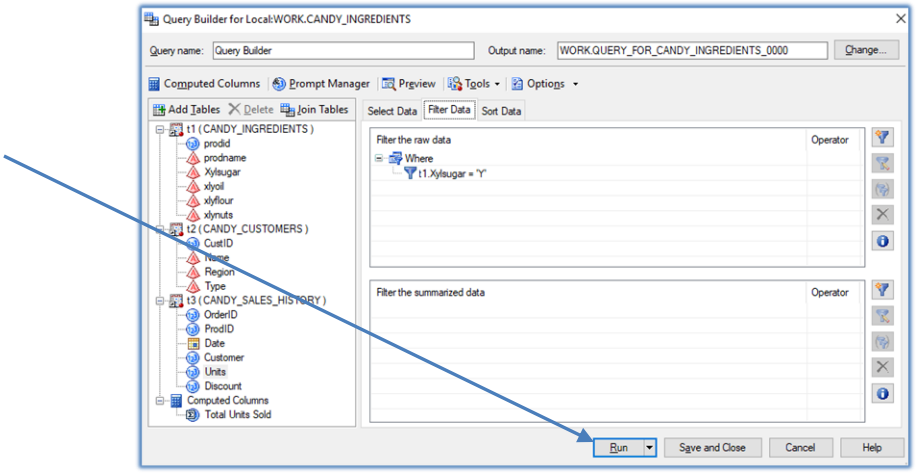


Figure 23 - New Filter Panels 3 & 4

22. Click the Run button to test out the query.



23. A listing of Companies, products and total units for Companies that have purchased products with xlysugar as an ingredient is the final output.
24. Click on the Code tab and view the code that EG created to accomplish this join.

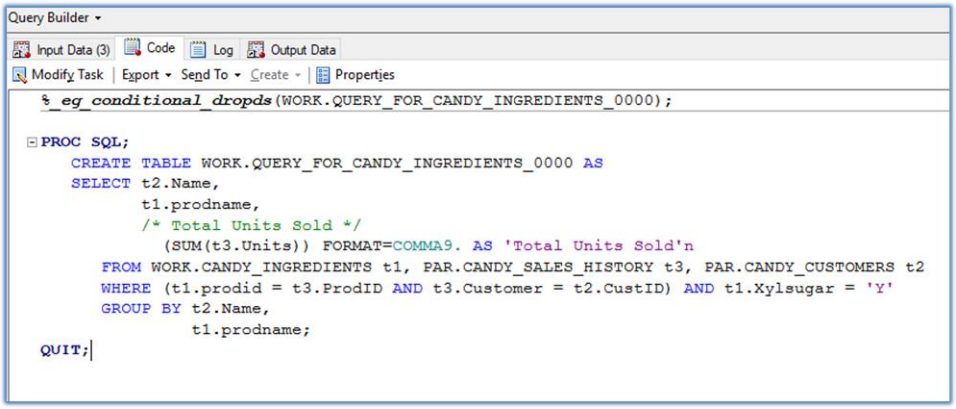


Figure 24 - Query Code

This code can now be copied and used in other programs. Changes to the query itself can be made by selecting the **Modify Tasks** button in the Code, Log, or Output Data window

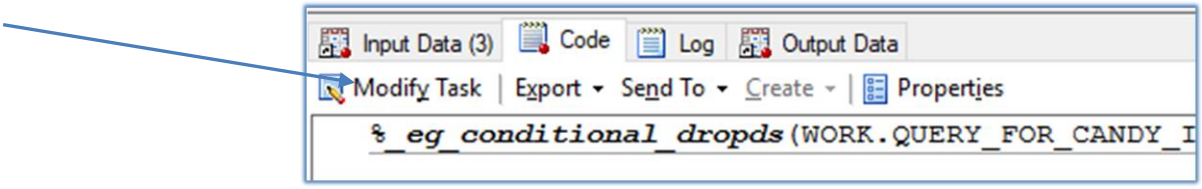


Figure 25 - Modify Task Button

STRATEGIES FOR EG TASKS

EG has incorporated a variety of tasks available. The tasks in EG are grouped by function: manipulate data (Data), describe data, graph data and analyze data. Each group has a multitude of options.

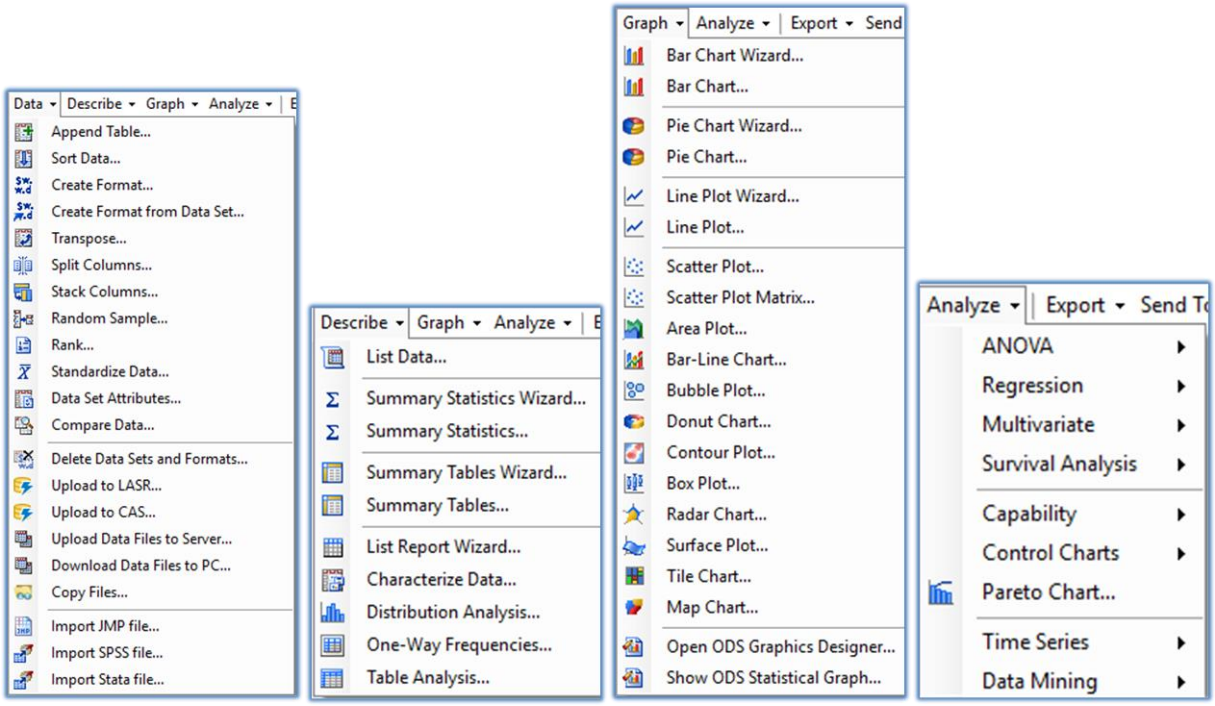


Figure 26 - EG Tasks

Tasks can be useful in many ways: when the code for a specific procedure isn't remembered, to do a quick analysis of data, or my favorite, beg, borrow and steal with pride.

When the task menus are used to either manipulate, describe, graph, or analyze data, code is created. This code can then be copied and pasted into another program and modified if needed.

EG decides how best to do the tasks requested, but it does allow for customization using the menus and there is even a place to add custom code to a task. Let's go through an example of how it can be helpful:

1. Open the sashelp.cars data set.
2. Use the Graph menu and select Pie Chart wizard.

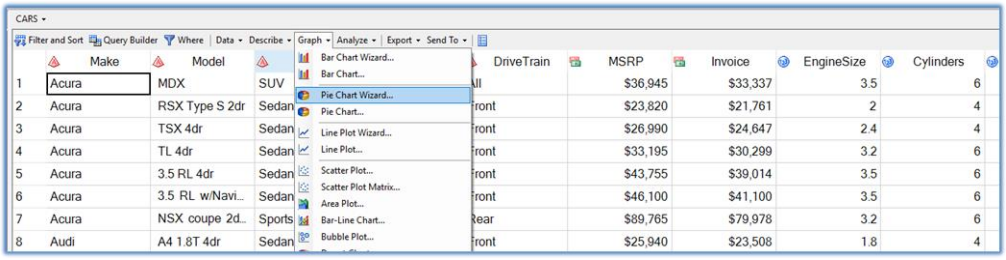


Figure 27 - Pie Chart Wizard Selection

3. Verify the Library and data set are the correct ones.

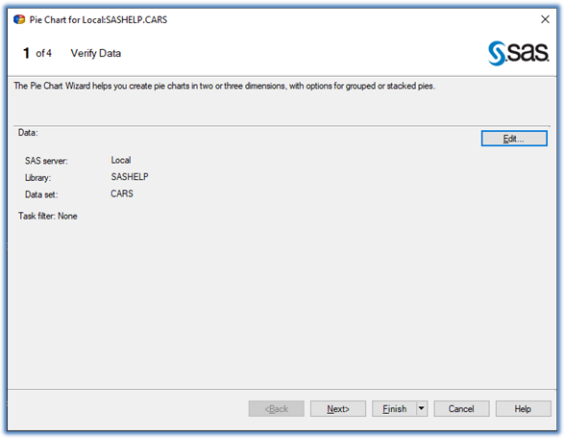


Figure 28 - Pie Chart Wizard Panel Step 1 of 4

4. Choose Make for the slice and (Frequency) for Slice Size (defaults).

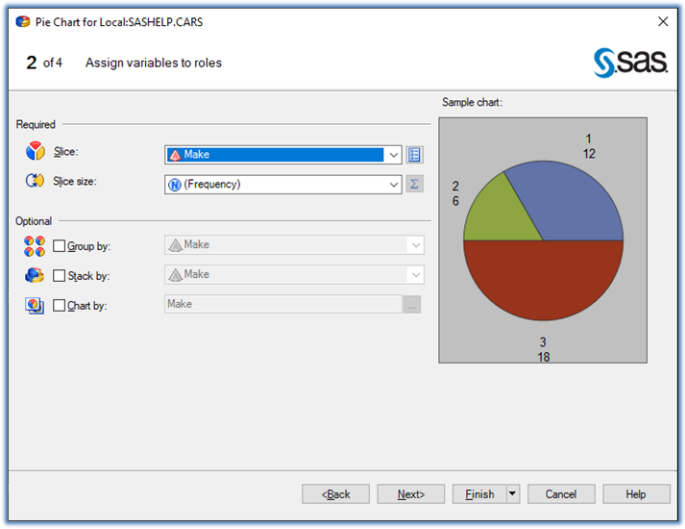


Figure 29 - Pie Chart Wizard Step 2 of 4

5. Choose the 3D chart, if desired.

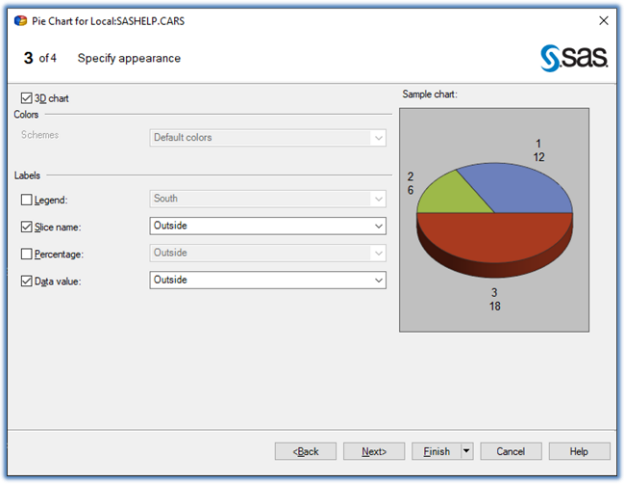


Figure 30 - Pie Chart Wizard Step 3 of 4

6. Notice the values in the Graph and Footnotes fields. These will populate the graph properties.

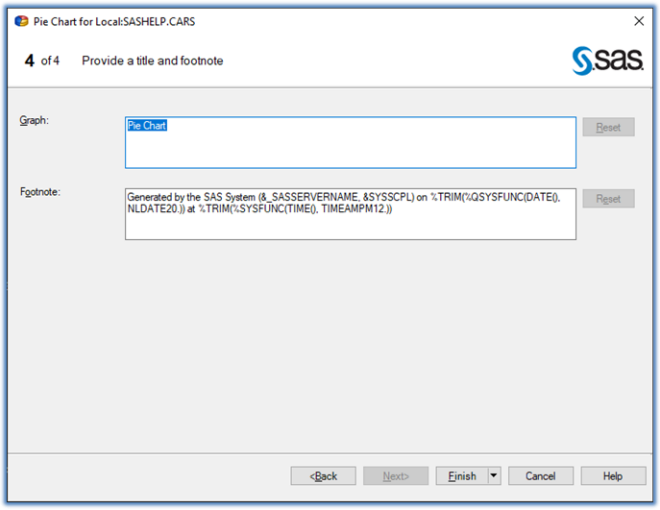


Figure 31 - Pie Chart Wizard Step 4 of 4

7. Observe the output from the Pie Chart Wizard Task.

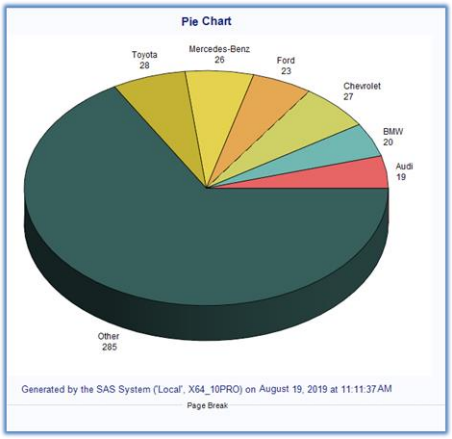


Figure 32 Pie Chart Wizard Results

8. Click on the Code window and select the code starting with the Proc SQL statement

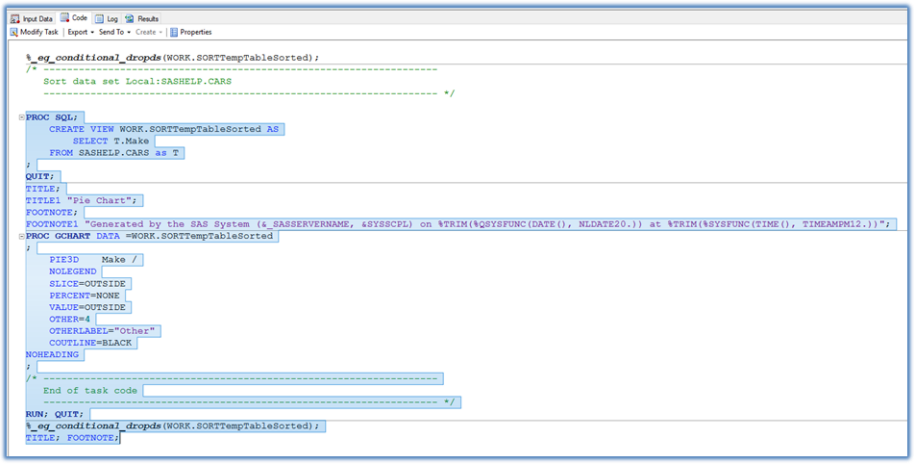


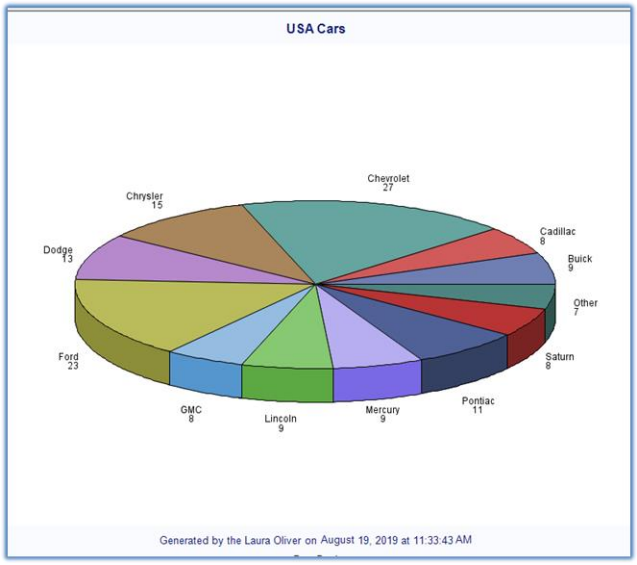
Figure 33 - Code Created From Task

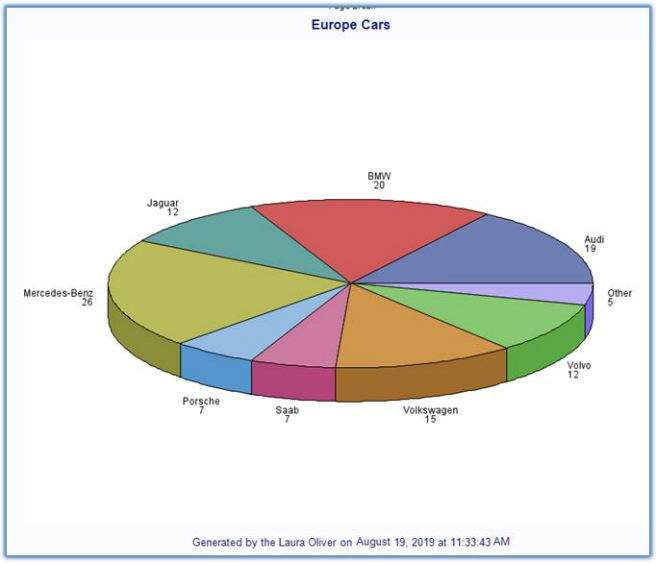
9. Paste the code into a new program window.

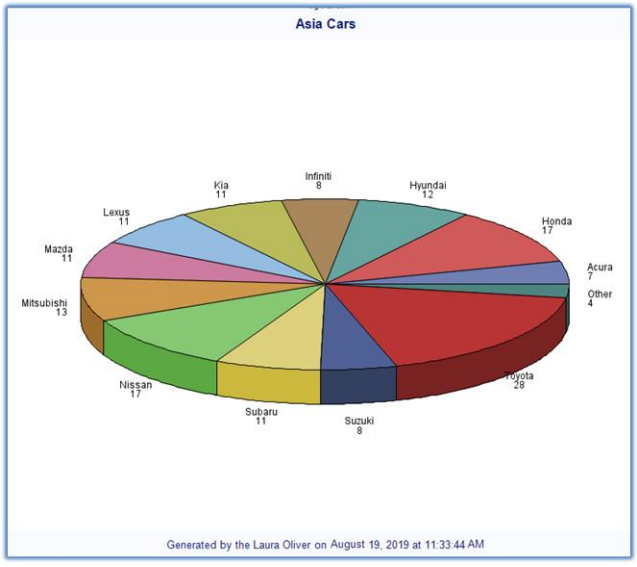
10. Modify the code to create a macro that will produce 3 pie charts, one for each region. Modify the Title and Footnote as desired.

```
%macro cars(type);
  PROC SQL;
    CREATE VIEW WORK.&type._car AS
      SELECT T.Make
         FROM SASHELP.CARS as T
           where origin = "&type."
     ;
  QUIT;

  TITLE;
  TITLE "&Type. Cars";
  FOOTNOTE;
  FOOTNOTE1 "Generated by the Laura Oliver ";

  PROC GCHART DATA = WORK.&type_car
  ;
      PIE3D     Make /
         NOLEGEND
         SLICE=OUTSIDE
         PERCENT=NONE
         VALUE=OUTSIDE
         OTHER=4
         OTHERLABEL="Other"
         COUTLINE=BLACK
           NOHEADING
    ;
  RUN;
  QUIT;
  TITLE;
  FOOTNOTE
%mend cars;

%cars(USA);
%cars(Europe);
%cars(Asia);
```

11. Run the new code and view the output.







CONCLUSION

As you can see, SAS has incorporated lots of short cuts into EG. Not only does it mean that you can gather information without having to write code to get it, but it can be used to create code for use in other programs. Both the short cuts and the code generating abilities make it easier to use EG for both the occasional user and the power developer.

CONTACT INFORMATION

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

Laura J Oliver
Laura.oliver@experis.com
www.experis.com