

Using SAS to Create Presentation Quality Spreadsheets in Excel

By Joyce R. Hartley, Infineon Technologies – Richmond

Abstract

How often have you been asked to produce a report that has subtotals here, grand totals there, ratios two lines down, bold this label, change that font... in short, a report better created in Excel? But all your data are in SAS data sets? Using PC SAS, DDE (Dynamic Data Exchange), and a little bit of cleverness, it's now easier than ever to produce nicely formatted Excel spreadsheets that are populated with SAS data. This technique is particularly useful for the report that needs to be repeated for multiple units (i.e. departments), where each sheet needs to be customized. An intermediate skill level in both base SAS and Excel is recommended.

Background

Imagine a project that requires you to produce displays with fairly similar data, grouped by management unit or division, but with slightly different categories on each display. And you want your lines in a specific order, with subtotals and percentages interspersed throughout the display. DATA _NULL_ is a fairly good option, although it can be rather tedious to go through all the iterations, especially when it comes to handling missing data. For example, a university budget office might require analysis of data by "budget unit" or "school" categories. These categories will be similar in some ways—they would all have budget for salaries—but very dissimilar in other ways. A School of Medicine would have a rather different budget from Facilities Management, as would a School of Business from Academic Technology. Wouldn't it be nice to write a single program to handle all the various cases and be able to create a spreadsheet with special formatting (fonts, headers, etc.) as well?

Method

The first step is to create an Excel template that is later copied as a blank "form" to be completed by the inclusion of your data output from SAS. This template needs to include all special formatting, formulae, and links to the worksheet that will contain the SAS data. Once you've created the template, the next step is to get your SAS data in order, all in one dataset, in the final form to be output to the Excel file. Then the program will execute the command to open Excel, send the SAS data to the workbook, and save and close the workbook.

Create an Excel Template

The first step in this process is to create your Excel template. This file needs to include all possible rows and columns that would appear on any of the spreadsheets that you plan to populate. These will simply be hidden in the spreadsheets where they are not applicable. Leave all the data cells blank, but apply any desired formatting to those cells; the data that come from SAS will be unformatted. Also select any page breaks, if applicable. My sample template includes rows for resource data such as degrees conferred and headcount; these rows would be hidden for a budget unit such as Facilities. Because my spreadsheet includes subtotals, percentages, and ratios, these formulae were also entered in the template file.

Create a second sheet in your workbook; keep the default sheet name of Sheet2. This is the sheet that will contain the data that are output from SAS, but it will be hidden from the user. Decide in what row and column your data will begin. Plan to populate all the cells in a rectangular area to match corresponding area in your main spreadsheet. That is, if you have blank columns as spacers in your main sheet, allow for blank columns in your SAS

Tools > Protection > Protect Workbook to protect the workbook itself (so that no one can unhide Sheet2). SAS can still write to the hidden sheet, but without the password, changes can't be made in Excel.

Create and Output the Data

Your SAS data may be coming from several different datasets. The goal is to get all the data combined into one dataset, ordered in such a way that there is a single sort column and the variable names, regardless of what kind of data they contain, all reference the columns in which

[illegible]

they will be placed. In my example, my sort column is called "order", and my column variable names are c95, c96, c97, c98, c99, and c00. Also include a dataset like this one when you combine all your data, to ensure that every row in your template Sheet 2 will get populated:

```
data dummy;
  do i=1 to 30;
    order=i;
    output;
  end;
run;
```

I try to match rows to rows in my two spreadsheets, that is, row 6 to row 6, which often leaves space in the first few rows for label variables that can be linked to the main sheet as well. A main title could be output to row 1, so that the titles don't have to be typed into each individual spreadsheet.

Putting It All Together

The final step in the process is for your program to send the command to open Excel, create a filename to write the data to Sheet2, and use DATA _NULL_ to populate the sheet, and then close Excel. If you are doing multiple sheets, you can enclose part of the code within a macro, passing the different filenames with each macro call. Then you get to sit back and watch while SAS fills in all your spreadsheets for you.

In order to send the command to open Excel, you need to know what directory excel.exe is in. I do this by exiting to the DOS prompt, and poking around until I find the right directory. It is usually in something like c:\Program Files\Microsoft Office\Office\Excel, but you need to find the right 8-character name for the directory, i.e. c:\progra~1\micros~4\office\excel. The code to open Excel will look something like this:

```
options noxwait noxsync;
/* this opens Excel */
x c:\progra~1\micros~4\office\excel';
```

```
data _null_;
  x=sleep(2);
/* this pauses SAS to
   give Excel time to open */
run;
```

Before submitting this code, be sure that Excel is not already open.

Here's the code to create the needed filenames and open the proper workbook:

```
filename cmds dde 'excel|system';
%let newfile=c:\my documents\my sas
files\lv8\ssupaper\wkb2.xls;
```

```
/* create macro var with name of
   excel file, since you can't use
   macro vars directly w/in an OPEN
   statement w/DDE */
```

```
data _null_;
  call symput
    ("exdata","||"[OPEN("||
    "&newfile"||")"]||""");
run;
```

```
/* open the file */
data _null_;
  file cmds;
  put &exdata;
run;
```

Next, create a filename to reference sheet2 and specify the rows and columns that you will be writing to. In the example below, I've created an additional variable called sp (for space) that simply writes blanks to map to my blank columns in my template worksheet.

```
/* output data to the temp1data
   worksheet */
filename worksht dde
  'excel|Sheet2!r1c1:r30c15' notab;
data _null_;
  set temp1data;
  file worksht;
  sp=' ';
```

SAS Output formatted in Excel

Anywhere University School of Marine Biology

	<u>1994-95</u>	<u>1995-96</u>	<u>1996-97</u>	<u>1997-98</u>	<u>1998-99</u>	<u>1999-00</u>	<u>6-Yr Ave.</u>
Headcount	123	118	120	124	129	132	124
Degrees Conferred	20	21	18	19	20	23	20
Annual Credit Hours	814	850	864	833	861	851	846
Expenditures by Ledger							
E&G	\$2,415,339	\$2,538,813	\$2,242,864	\$2,453,342	\$2,864,638	\$2,776,644	\$2,548,606
Sponsored Programs	13,172	29,191	3,650	2,959	17,125	50,592	19,448
Facilities	28,352	13,773	185,578	41,207	80,701	88,151	72,960
Local Funds	435,104	597,710	783,491	490,266	356,959	475,495	523,154
State Funds	263,858	301,926	161,254	301,760	314,551	432,642	295,999
Aquarium Services	1,329	2,847	6,247	80,071	85,492	15,378	31,894
Grand Total	\$3,157,153	\$3,484,160	\$3,383,084	\$3,369,605	\$3,719,466	\$3,838,903	\$3,492,062
Expenditures by Category							
Teaching/Research	\$1,142,762	\$1,084,864	\$870,751	\$889,878	\$1,233,944	\$1,188,039	\$915,748
Graduate/Teaching Assistant	236,388	319,235	338,948	473,161	494,795	505,053	338,226
Adjunct Faculty	798,249	792,401	738,633	872,478	1,000,191	1,068,892	752,978
Other Personal Services	301,549	402,256	400,039	378,054	297,886	273,433	293,317
Total Personal Services	\$2,478,948	\$2,598,756	\$2,348,371	\$2,613,571	\$3,026,815	\$3,035,418	\$2,300,268
Contracts	61,235	52,252	127,088	89,780	86,463	102,247	74,152
Travel	129,476	283,483	352,524	125,657	96,183	104,361	155,955
Equipment	487,108	518,231	559,395	545,505	510,409	596,877	459,646
Other Nonpersonal Services	386	31,438	(4,295)	(4,907)	(404)	0	3,174
Total Nonpersonal Services	\$678,205	\$885,404	\$1,034,713	\$756,035	\$692,651	\$803,485	\$692,927
Grand Total	\$3,157,153	\$3,484,160	\$3,383,084	\$3,369,605	\$3,719,466	\$3,838,903	\$2,993,196
E&G Expenditures per Headcount	\$19,637	\$21,515	\$18,691	\$19,785	\$22,206	\$21,035	\$20,478
Total Expenditures per Headcount	\$25,668	\$29,527	\$28,192	\$27,174	\$28,833	\$29,083	\$28,079
% of E&G Expenditures to Total	76.50%	72.87%	66.30%	72.81%	77.02%	72.33%	72.98%

```

put order '09'x sp '09'x
      c95 '09'x sp '09'x
      c96 '09'x sp '09'x
      c97 '09'x sp '09'x
      c98 '09'x sp '09'x
      c99 '09'x sp '09'x
      c00 '09'x sp '09'x avg ;
run;

```

And finally, close Excel:

```

/* close Excel */
data _null_;
  file cmds;

```

```

put '[SAVE()]';
put '[QUIT()]';
run;

```

Finishing Touches

Once all your workbooks are filled in, all you need to do is hide any rows that you choose to. You may find that some cells containing formulas come up with Excel errors, such as division by zero. In these cases, you can apply conditional formatting to these cells to replace the error with "N/A" or some other

indicator. It will probably take a few runs before you get your template file perfected, but once you do, you can copy it over for as many units as you need, and re-use it as often as you need. Nicely formatted monthly or weekly reports can be produced in minutes now, just by copying over the template file and changing the input and output filenames. The possibilities are endless.

Acknowledgments

Thanks to Mike Newsome for encouraging me to push my limits, and to Andre Walker, Jim Moyar, Bennie Fiol, Delores Anderson, Greg Vaeth and Michelle Vucci for all their help.

SAS is a registered trademark of SAS Institute, Inc.

Excel is a registered trademark of Microsoft Corporation.

Contact Information

The author can be reached via e-mail at joyce.hartley@infineon.com, or by mail at
Joyce R. Hartley
Infineon Technologies Richmond
6000 Technology Boulevard
Sandston, VA 23150
(804)952-8262