Paper 060-2007

Excelling with SAS®

Faron Kincheloe, Baylor University, Waco, TX

ABSTRACT

You want to present your data using the formatting and power of an Excel spreadsheet. However, you want to use SAS® to extract and periodically update the data on which the spreadsheet is based. Using the improved functionality of the export procedure in SAS® 9 to write to a specific sheet, you can have your format and update it, too.

INTRODUCTION

This paper describes the process of using the SAS® export procedure to supply the underlying data to a spreadsheet with the desired formatting including graphs and calculations.

THE PROBLEM

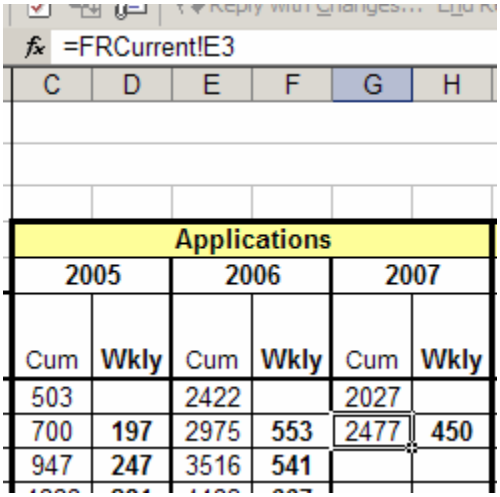
Programs have been written using SAS® to extract information from a transactional system. The programs are scheduled to run automatically; some nightly and others weekly. The end users desire to have the information available in Excel spreadsheets for a number of reasons. Excel is widely used within the organization. Users can use Excel to perform additional calculations with the data including "what if" scenarios and readily create their own charts and graphs. In some cases spreadsheets have already been created showing periodic performance to date. When the results for a new period are finalized, the user updates the spreadsheet manually by taking the data from SAS® output.

Prior to SAS® version 9, the export procedure could only write to a single sheet in an Excel file. Each time the export procedure was used it replaced the previously existing Excel file. This would destroy any calculations or formatting that had been done since the initial export. This limitation created somewhat of a "one-shot" process that could not be easily updated without a certain amount of manual effort.

THE SOLUTION

The export procedure in version 9 adds the capability to create or replace a specific sheet in an Excel workbook without affecting any other sheets. The exported data is still in the form of a raw unformatted table but the ability to write to a specific sheet provides a method for circumventing the problem described above. The graphic below shows the tabs of an Excel workbook that is now automatically updated using the export procedure.

This workbook was previously updated manually every weekend. The yellow FR Trends tab is the sheet that contained tabular data that was appended with a new section of result each week. The four tabs to the left of the yellow tab are sheets that contain charts that graphically represent the data in various columns of the FR Trends table. The red FRCurrent tab is a sheet that was originally created by the SAS® export procedure and is updated by the same procedure every Friday night. The original numbers in the FR Trends sheet were replaced with references to cells in the FRCurrent sheet. The FR Trends sheet displays all of the data from the FRCurrent sheet as well as some additional columns that are calculated based on FRCurrent data. Each time the spreadsheet is opened, all of the formulas are recalculated so that the most up-to-date information is displayed. The image below shows an example of a reference formula for one of the cells in the FR Trends sheet.



The first step in an integration project such as this is to organize the SAS® data in a layout that is conducive to being referenced within an Excel workbook. The layout needs to remain constant over time so that the Excel references will be predictable. Otherwise, data can end up in the wrong cells and defeat the purpose of the project.

In this example, proc report is used to create summary values in a dataset that is then exported to our Excel workbook. The code below is an excerpt from the report procedure showing how the output dataset is created.

```
proc report data=all nowd headline headskip split='/' out=FRappcounts;
where apcode='F';
   columns week_num weekend
           ('Applied' app1 app2 app3)
           ('Accepts' acpt1 acpt2 acpt3)
           ('Deposits' dep1 dep2 dep3)
           ('Net Deposits' nd1 nd2 nd3)
            ('Registered' reg1 reg2 reg3);
   define week_num / group 'Week';
   define weekend / group format=date5. 'Ending';
   define app1     / sum "&firstyear" ;
```

The column headings are not necessary but are provided to assist in troubleshooting as the layout is prepared for the first time. Any procedure or data step code can be used as needed to provide the layout for the data to be exported to Excel.

The code below exports the data to the FRCurrent sheet without affecting any other sheets in the Excel file.

```
/* Write data out to excel file */
PROC EXPORT DATA= WORK.frappcounts
            OUTFILE= "F:\PortalDownloads\Trends.xls"
            DBMS=EXCEL REPLACE;
     SHEET="FRCurrent";
RUN;
```

There are several parameters that control the export procedure. DATA defines the source of the data to be exported. OUTFILE supplies the name and the path of the destination Excel file. Since the export procedure can write to a number of formats the DBMS parameter instructs the procedure to write to the Excel format. The REPLACE keyword is essential as it instructs the procedure to update the sheet if it already exists. If the sheet does not exist it will be created. Finally, the SHEET parameter defines the specific sheet to which the data will be exported within the Excel file. Pay special attention to the placement of semicolons in the example above as this proved to be a troublesome detail in the original development of this project.

OTHER ISSUES

The SAS® default of using periods for missing data may not be desirable within Excel. This can be changed to blanks by using the following option statement at the beginning of the program:

```
options missing=' ';
```

In the original spreadsheet, it was easy to locate the current time period in the tabular data by scanning down the column to find the last item of data. However, all of the cells relating to future time periods now contain zeroes since cells that were once populated on a week-by-week basis now contain a reference to another sheet. Even though the cell is empty in the FRCurrent sheet which is updated by SAS®, the reference will still return a value of zero. The zeroes can be suppressed using a custom cell format. First, highlight all of the cells for which zeroes are to be suppressed. Select Format then choose Cells. Make sure the Number tab is selected. In the Category pane, select Custom. Clear any characters from the box labeled Type:. In that box enter the following string of numbers and characters 0;-0;;@ then click OK. The main drawback of using this format is that any zero value from a computation will be replaced with a blank.

The zero values also created a problem with the graphs. Keep in mind that when the workbook was created, graphs were based on the tabular data from the FR Trends sheet. Since future time periods would be empty, the series on the graph would simply stop at the current time period. Now, the future time periods all contain zeroes so the series continues but drops to a flat line at zero beyond the current time period. This problem was resolved by simply changing each series in the graph to reference the SAS® data in FRCurrent.

CONCLUSION

You can use the export procedure to write SAS® data to a specific sheet within an existing Excel file. The other sheets use references to point to the SAS® data. This allows you to retain all of your calculations and formatting. By using the power of both products, you can literally excel with SAS®.

ACKNOWLEDGEMENTS

CONTACT INFORMATION

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

Faron Kincheloe
Baylor University
One Bear Place #97032
Waco, TX 76798
Phone: (254) 710-8835
Email: Faron_Kincheloe@baylor.edu