Paper 207-2012

Getting Started with ODS: Generating Formatted Reports Using the ExcelXP Tagset

Allan Del Rosario, Marriott International, Bethesda, MD
Jennifer Mefford, Marriott International, Bethesda, MD

ABSTRACT

In the business world, reporting is a key component in presenting data to the user. Using the power of the SAS® ODS ExcelXP tagset destination to customize report structure and format is one method of report creation. This method is particularly well-suited to automating the production of multiple Excel workbooks that are nicely laid out for users.

This paper shows a simple approach to getting started with writing ODS using the ExcelXP tagset. It demonstrates how to use the ODS style options to generate formatted reports in Excel directly from SAS®, and demonstrates how to easily add iterative DO loops and macro variables to automatically generate multiple formatted reports.

INTRODUCTION

There are several techniques that can be employed to transfer data from SAS® to Excel. Using the Output Delivery System (ODS) with the ExcelXP tagset destination is especially well suited to automatically generating Excel reports. All of the options to control formatting in the ExcelXP tagset destination, the ability to display multiple tables on one worksheet, and the ability to create multiple worksheets give the developer a high degree of flexibility in creating Excel reports directly from SAS®. This flexibility means that the developer does not need to open the Excel workbook and format it, nor does the developer need to create an Excel template.

By using loops in combination with the ODS and the ExcelXP tagset, the developer can quickly and easily generate multiple formatted reports. This is particularly useful if you need to create individual reports for multiple business locations, employees, or products.

REQUIREMENTS

The techniques described in this paper utilize the following software:

- Base SAS® 9.1.3 or later
- Microsoft Excel 2002 or later
- SAS® ODS ExcelXP tagset

GETTING STARTED WITH ODS

Creating a Template

Before you begin writing the code to output your report, you will probably already have some idea of what you would like the final report to look like. You can create a custom style template for your report using the TEMPLATE Procedure. A custom style template allows you to control nearly every aspect of your report from the background color to the page layout to the cell formats. SAS® has several styles built in that you can use. The easiest way to create a custom style template is to start with an existing style that is close to what you want, and then modify the components as needed to achieve the settings you desire.

You can use this code to display a list of the built-in style templates in the Output window:

```
PROC TEMPLATE;
  List styles;
RUN;
```

If you would like to see the code for any of the built-in templates, you can do that from the Display Manager. This can be very helpful if you would like to see how to change a certain component. To see the style template code from the Display Manager:

1. From the menu bar at the top of the screen, click on View then select Results
2. From the Results window, right click on Results and select Templates
3. Expand the Sashelp.Tmplmst with the "+" symbol
4. Click on Styles to display a list of the built-in style templates
5. Double clicking on any of the listed styles will open the code

Below is an example of PROC TEMPLATE code. A new style template "VwaExcel" is being created. In this code, we have chosen to start with the built-in style template "Meadow" and then modify some of the components such as the background and foreground color and some of the font attributes.

```
PROC TEMPLATE;
  define style Styles.VwaExcel;
   parent = styles.Meadow;
   style
      SystemTitle from SystemTitle /
      background = white
      foreground = black
      font_style = Roman
      font_weight = Bold
      font_size = 14pt
      font_face = "Marriott Light";
   END;
RUN;
```

It should be noted that creating a style template is not required. If you would simply like to use one of the built-in style templates you do not need to use the PROC TEMPLATE code.

Setting the File Name, Style and Options for the Workbook

Now that you have created a custom template with the PROC TEMPLATE or decided on a built-in style template, it's time to define the ODS destination. "Listing" is the default ODS destination so first we want to close "listing". Next, we set the ODS destination to the ExcelXP tagset. In the same line we need to establish the pathway and name for the file being created and set the style we want to use. As you have seen, the style determines the appearance of the output. In this case, we are using the style template we created in the PROC TEMPLATE, "VwaExcel."

We have also added a few of the options available. There are more than 50 options that can be used to format the spreadsheet. These include such settings as page orientation, height, headers, footers, and sheet names. A quick reference for the ExcelXP tagset options can be found in the knowledge base area of the SAS.com website.

```
ODS listing close;
ODS tagsets.ExcelXp
   file='C:\ODS_Example.xls'
   style=VwaExcel
   options    (embedded_titles='yes'   suppress_bylines='yes'
               Orientation='Landscape' sheet_name="Boston Marriott"
               fittopage='yes'         Pages_FitWidth ='1'
               Pages_FitHeight = '1'   frozen_headers='6'
               center_horizontal='yes' sheet_interval='none'
               Row_Repeat = '6');
```

Adding Titles and Footnotes

Titles and footnotes are global statements in SAS®, so you can set them anytime, as long as they come before the PRINT procedure.

```
title1 'Hotel Performance Report';
title2 "Segment Production YTD";
title3 "Property: Boston Marriott";
title4 "Timeframe: 2012";
footnote 'MARRIOTT CONFIDENTIAL AND PROPRIETARY INFORMATION';
```

Writing Data to Excel

The ODS ExcelXP tagset destination can be used to export the results of PROC PRINT, PROC REPORT or PROC TABULATE. In this paper, we have chosen to use PROC PRINT.

In the example below, the style options can be applied to the sheet by using them with the PROC PRINT or they can be applied to an individual variable by using them in the VAR statement.

Applying styles to the headers in the example below sets the justification, font size, font face and border widths.

The tag attribute (TAGATTR) is used to display the variables in Excel-defined formats. Tag attributes generally correspond to the number type you can see in the cell format box in Excel when you select “custom” as the number type. For example, 'format:0.0%_);[Red]-0.0%' writes the variable as a percentage and applies a red font color for negative values. The 'format:#,##' formats the numeric data to display commas.

Because many of the labels for our column headers are longer than will fit within the cell width allotted, the text needs to wrap to the next line. The SPLIT command tells SAS what character you will use to indicate that the text should wrap to the next line at that point. In this example a “*” is used (split = '*').

```
PROC PRINT data=example1 noobs label
         style(header)={just=center          font_face = "Marriott Light"
                        font_size=11pt       bordertopwidth=1
                        borderbottomwidth=1  borderleftwidth=1
                        borderrightwidth=1}
         style(data)={font_size=10pt         font_face = "Marriott Light"
                        bordertopwidth=1     borderbottomwidth=1
                        borderleftwidth=1    borderrightwidth=1}
         split='*';
   where hotel ="Boston Marriott";

   label SEG='Segment'
         RNTY='Roomnights *This Year'
         RNLY='Roomnights *Last Year'
         RNCHG='Roomnight *Change'
         RVTY='Revenue *This Year'
         RVLY='Revenue *Last Year'
         RVCHG='Revenue *Change';

   VAR SEG /style={cellwidth=1in};
   VAR RNTY RNLY /style={tagattr='format:#,##' just=center cellwidth=1in};
   VAR RNCHG /style={tagattr='format:0.0%_);[Red]-0.0%' just=center
                  cellwidth=1in};
   VAR RVTY RVLY /style={tagattr='format:$#,##0_);($#,##0)' just=center
                  cellwidth=1in};
   VAR RVCHG /style={tagattr='format:0.0%_);[Red]-0.0%' just=center
                  cellwidth=1in};
RUN;
```

Close the ODS tagsets.ExcelXP

Just as we closed "listing" as the ODS destination and set it to "tagsets.ExcelXP" at the beginning of our ODS code, we now need to close the "tagsets.ExcelXP" destination and set it back to "listing." Closing the ExcelXP tagset is essential because it closes and releases the file that has been created in the PROC PRINT so that it can be opened with Excel.

```
ODS tagsets.ExcelXP close;
ODS listing;
```

View Output

Hotel Performance Report
Segment Production YTD
Property: Boston Marriott
Timeframe: 2012

Segment	Roomnights This Year	Roomnights Last Year	Roomnight Change	Revenue This Year	Revenue Last Year	Revenue Change
Weekday	25,640	23,547	8.9%	$6,410,000	$5,769,015	11.1%
Weekend	5,923	5,896	0.5%	$1,036,525	$1,002,320	3.4%
AAA	7,572	8,579	-11.7%	$1,249,380	$1,312,587	-4.8%
Senior	2,885	2,567	12.4%	$476,025	$415,854	14.5%
Total	42,020	40,589	3.5%	$9,171,930	$8,499,776	7.9%

GENERATE MULTIPLE INDIVIDUAL REPORTS

Perhaps your data set contains data for multiple regions or multiple accounts and you would like to produce a report for each region or account that only contains the data for that region or account. Creating a single report is great, but the real advantage of using the ODS ExcelXP tagset destination is that you can easily add macro variables and an iterative DO loop to produce multiple individual reports.

Create List of Individual Reports

The first step is to create a list of the distinct variables you want to use to create your reports. In our example, we are going to produce one report for each hotel, so we have created a data set called "num_hotels" that is a list of the unique hotel names in the "example1" data set.

Second, we need to count how many distinct values there are. We will need this later to tell SAS® how many times to loop through the ODS code so that one report is created for each hotel. We have created a macro variable called "numcmp" with the total number of hotels in the "num_hotels" data set.

The third step is to assign a macro variable to each hotel name. To make things simple we have chosen to assign our hotels to the macro variables starting with E1 and continuing sequentially to E&numcmp (remember "numcmp" represents the total number of hotels).

We have now dynamically created a list of every hotel that we need to produce a report for and assigned those hotels to a macro variable that can easily be incremented in a DO loop. When the number of hotels in our data set changes in the future we will not need to manually update anything in our code: it is already set-up to adjust for this.

```
PROC SQL;
  create table num_hotels as
  select distinct hotel
  from example1;
QUIT;
```

```
DATA _null_;
  set num_hotels end=last;
  if last then call symput("numcmp",compress(put(_n_,12.)));
RUN;

%PUT &numcmp;

PROC SQL NOPRINT;
  select distinct hotel
   into :E1 - :E&numcmp
   from num_hotels;
QUIT;
```

Add a Loop to the ODS Code

The final step is to add the loop to our code. Using our "numcmp" macro variable we can set the number of iterations. The "do" statement will go right before the ODS "listing" is closed and the "end" will go right after the ODS destination is reset to "listing" (after the PROC PRINT).

```
%DO I=1 %TO &numcmp;
%END;
```

FINAL CODE

In the final code, we have replaced hard coded names such as file names, sheet names and titles with the macro variables representing the hotel name (`&&E&I`) to dynamically place the correct hotel name in each loop. We have also replaced hard-coded timeframes with a macro variable (`&curyr`) to automatically place the correct year so that the code does not have to be updated when the year changes.

Note: Any name that you plan to use in a file name will need to satisfy the restrictions on length and allowed characters that are imposed by your file system.

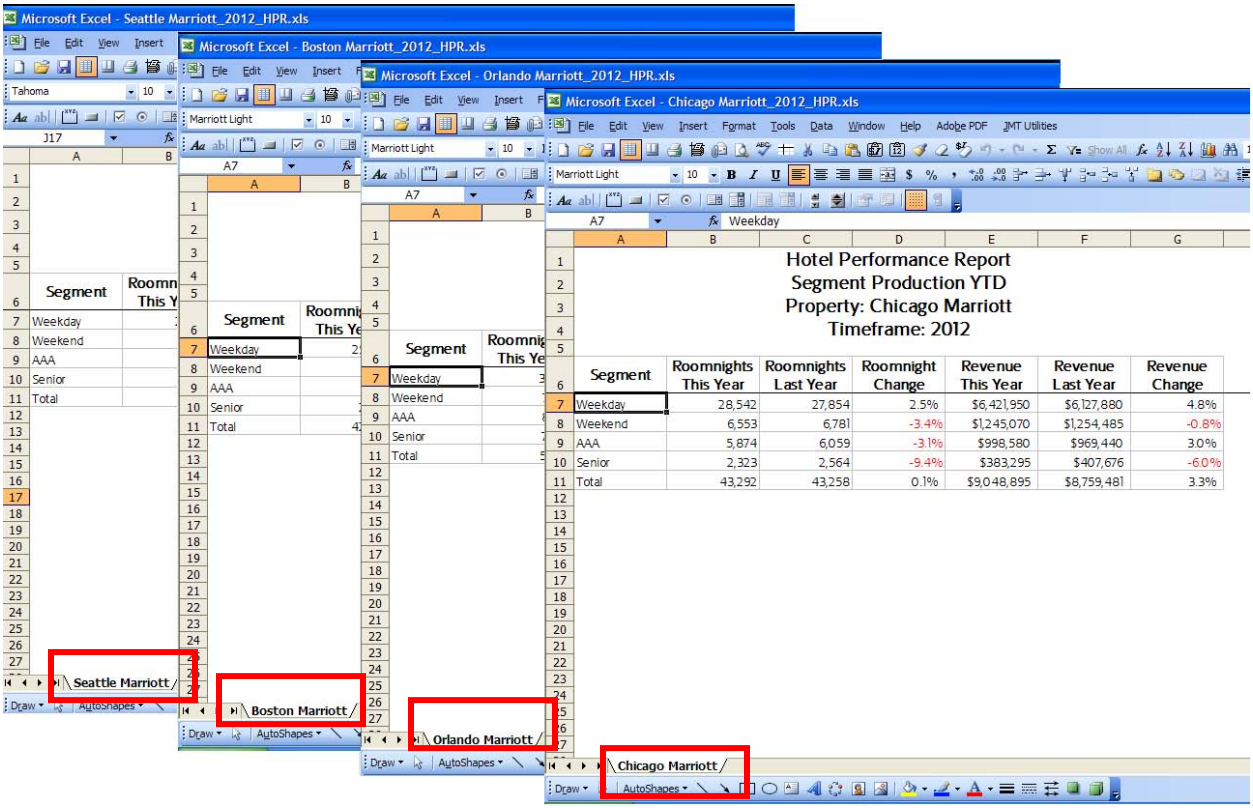
```
PROC SQL;
  create table num_hotels as
  select distinct hotel
  from example1;
QUIT;

DATA _null_;
  set num_hotels end=last;
  if last then call symput("numcmp",compress(put(_n_,12.)));
RUN;

%PUT &numcmp;

PROC SQL NOPRINT;
  select distinct hotel
  into :E1 - :E&numcmp
  from num_hotels;
QUIT;

PROC TEMPLATE;
  define style Styles.VwaExcel;
   parent = styles.Meadow;
   style
   SystemTitle from SystemTitle /
      background = white
      foreground = black
      font_style = Roman
```

```
      font_weight = Bold
      font_size = 14pt
      font_face = "Marriott Light";
   end;
RUN;

options nomprint nosymbolgen nomlogic nonotes nomacrogen;
%MACRO OUTDATA;

%DO I=1 %TO &numcmp;

%let xlssheet=%STR(C:\Documents and Settings\Test\);

ODS listing close;
ODS tagsets.ExcelXp
   file="&xlssheet.&&E&I.._&curyr._HPR.xls"
   style=VwaExcel
   options    (embedded_titles='yes'   suppress_bylines='yes'
               Orientation='Landscape' sheet_name="&&E&I"
               fittopage='yes'         Pages_FitWidth ='1'
               Pages_FitHeight = '1'   frozen_headers='6'
               center_horizontal='yes' sheet_interval='none'
               Row_Repeat = '6');
options center LeftMargin='1in' RightMargin='1in' TopMargin='1in'
        BottomMargin='1in';
  title1 'Hotel Performance Report';
  title2 "Segment Production YTD";
  title3 "Property: &&E&I";
  title4 "Timeframe: &curyr";
  footnote 'MARRIOTT CONFIDENTIAL AND PROPRIETARY INFORMATION';

PROC PRINT data=example1 noobs label
         style(header)={just=center          font_face="Marriott Light"
                        font_size=11pt       bordertopwidth=1
                        borderbottomwidth=1  borderleftwidth=1
                        borderrightwidth=1}
         style(data)={font_size=10pt         font_face="Marriott Light"
                        bordertopwidth=1     borderbottomwidth=1
                        borderleftwidth=1    borderrightwidth=1}
         split='*';
   where hotel ="&&E&I";

   label SEG='Segment'
         RNTY='Roomnights *This Year'
         RNLY='Roomnights *Last Year'
         RNCHG='Roomnight *Change'
         RVTY='Revenue *This Year'
         RVLY='Revenue *Last Year'
         RVCHG='Revenue *Change';

   VAR SEG /style={cellwidth=1in};
   VAR RNTY RNLY /style={tagattr='format:#,##' just=center cellwidth=1in};
   VAR RNCHG /style={tagattr='format:0.0%_);[Red]-0.0%' just=center
               cellwidth=1in};
   VAR RVTY RVLY /style={tagattr='format:$#,##0_);($#,##0)' just=center
               cellwidth=1in};
   VAR RVCHG /style={tagattr='format:0.0%_);[Red]-0.0%' just=center
               cellwidth=1in};
RUN;
```

```
ODS tagsets.ExcelXP close;
ODS listing;
  %END;
%MEND;

%OUTDATA; /*Run the Macro*/
```

View Final Output

Using a combination of loops and macro variables with the ODS ExcelXP tagset destination, we have dynamically generated reports for multiple properties using our final code.



CONCLUSION

ODS ExcelXP is a great way to format the appearance of Excel reports directly from SAS®. This method provides a high level of flexibility and customization. Creating Excel reports directly from SAS® without needing to open Excel and make formatting changes often saves time and creates a more consistent end result, particularly when you need to generate multiple reports.

REFERENCES

Andrews, Rick. “Printable Spreadsheets Made Easy: Utilizing the SAS® Excel XP Tagset,” NESUG (2008), http://www.nesug.org/proceedings/nesug08/ap/ap06.pdf

DelGobbo, Vincent. “More Tips and Tricks for Creating Multi-Sheet Microsoft Excel Workbooks the Easy Way with SAS,” SUGI (2009), http://support.sas.com/resources/papers/proceedings09/152-2009.pdf

Gupta, Sunil K. “Using Styles and Templates to Customize SAS ODS Output,” SUGI 27 (2001), http://www2.sas.com/proceedings/sugi27/p007-27.pdf

Miralles, Romain. “Creating an Excel report: A comparison of the different techniques,” SUGI (2011), http://support.sas.com/resources/papers/proceedings11/074-2011.pdf

CONTACT INFORMATION

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

Jennifer Mefford
Marriott Corporate Headquarters
Department 55/958.03
10400 Fernwood Road
Bethesda, MD 20817
Phone: (301) 380-8485
Jennifer.Mefford@marriott.com

Allan Del Rosario
Marriott Corporate Headquarters
Department 55/958.03
10400 Fernwood Road
Bethesda, MD 20817
Phone: (301) 380-4057
Allan.DelRosario@marriott.com