Paper 1854-2014

Exporting formulas to Excel using the ODS ExcelXP tagset

Joseph K Skopic, Federal Government, Washington, DC

ABSTRACT

SAS® can easily perform calculations and export the result to Microsoft Excel in a report. However, sometimes you need Excel to have a formula or a function in a cell and not just a number. Whether it's for a boss who wants to see a SUM formula in the total cell or to have automatically updating reports that can be sent to people who don't use SAS to be completed, exporting formulas to Excel can be very powerful. This paper illustrates how, by using PROC REPORT and PROC PRINT along with the ExcelXP tagset, you can easily export formulas and functions into Excel directly from SAS. The method outlined in this paper requires Base SAS® 9.1 or higher and Excel 2002 or later and requires a basic understanding of the ExcelXP tagset.

INTRODUCTION

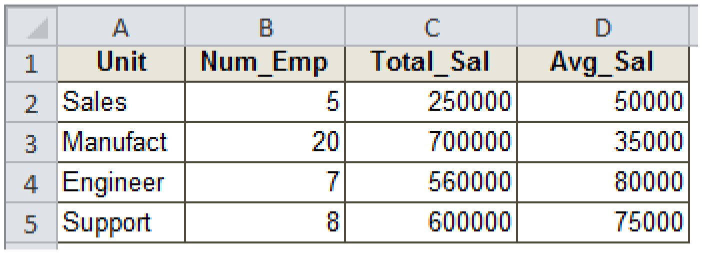
This paper will introduce using the ODS ExcelXP tagset to export formulas into Excel instead of just values from SAS using the REPORT and PRINT procedures. You may ask, "If SAS can perform all the calculations that I need, why I would want to export a formula into Excel and perform the calculation there?" There could be many reasons you want to do this including a supervisor who just likes to see the SUM formula in total lines or the need for a dynamically updating workbook. This second reason is the one that originally interested me in this subject, I needed to produce a series of reports on a bi-weekly basis that would be filled with to-date information which would then be sent to 15 operating divisions to be completed with projections and returned. I could have programmed an Excel Macro to fill in all the total lines so they would automatically calculate, but then I would have to remember to run the Macro on each workbook every two weeks. I hoped there was an easier way to do this using SAS and that's when I found the ExcelXP tagset.

SAMPLE DATA

For the purpose of this paper I put together a short example to illustrate the technique of exporting formulas into Excel. In the example there are four operating divisions and we are given the number of employees (Num_Emp) and the total salary of all the employees in each division (Total_Sal). We are then going to calculate the average salary (Avg_Sal) in each division and export the results into Excel.

```
DATA Staffing;
      INPUT Unit $ Num_Emp Total_Sal;
      DATALINES;
Sales 5 250000
Manufacturing 20 700000
Engineering 7 560000
Support 8 600000
;
RUN;

DATA Staffing;
      SET Staffing;
      Avg_Sal = Total_Sal/Num_Emp;
RUN;

ODS tagsets.ExcelXP PATH='C:\Users\Desktop' FILE='Staffing.xml'
      STYLE=analysis;

PROC REPORT DATA=Staffing NOWD;
      COLUMNS Unit Num_Emp Total_Sal Avg_Sal;
RUN;

ODS tagsets.ExcelXP CLOSE;
```

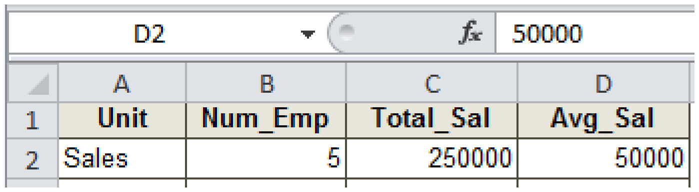
The first DATA step inputs the sample data, then in the second DATA step the average salary is calculated; finally using PROC REPORT and ODS the information is exported to Excel. This produces a very simple report that shows the input information along with the calculated average salary.



	A	B	C	D
1	Unit	Num_Emp	Total_Sal	Avg_Sal
2	Sales	5	250000	50000
3	Manufact	20	700000	35000
4	Engineer	7	560000	80000
5	Support	8	600000	75000

Display 1. Excel export #1

Looking at the average salary field in cell D2 we see that SAS exported the value calculated in the second DATA step, but what if you wanted to see the formula calculating the average and not just the value?



D2 fx 50000

	A	B	C	D
1	Unit	Num_Emp	Total_Sal	Avg_Sal
2	Sales	5	250000	50000

Display 2. Excel export #1 detail

NOTE: If you use an .xls or .xlsx file extension you may get a warning when opening the Excel file “The file you are trying to open is in a different format than specified…. Do you want to open the file now?” respond yes and it should not cause any issues. This is because the ExcelXP tagset actually creates a XML file even if you call for an .xls or .xlsx file in ODS.

BASICS OF EXPORTING FORMULAS

For the purpose of this example I am going to assume that you have a basic understanding of how the ExcelXP tagset works. The ExcelXP tagset is a very powerful tool which many options and attributes but the one I am going to use is the TAGATTR= style attribute. The TAGATTR= style attribute is used to send Excel instructions into cells.

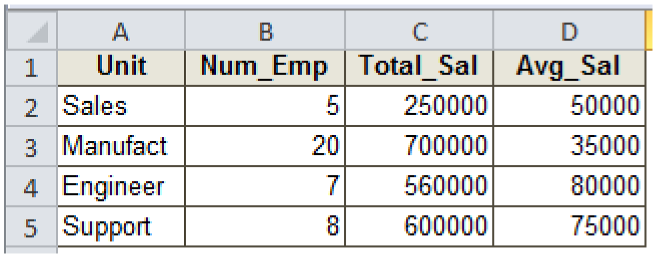
The syntax to export a formula using the TAGATTR= style attribute is “formula: ” after the equals with the formula you want exported placed after the colon before the closing quotation mark. In the example to export the average salary calculation as a formula I added the formula to calculate the average salary with the TAGATTR= style attribute.

```
tagattr="formula:RC[-1]/RC[-2]"
```

I then added this new code to the PROC REPORT using a DEFINE statement and STYLE option for the Avg_Sal variable and reran the code.

```
ODS tagsets.ExcelXP PATH='C:\Users\Desktop' FILE='Staffing.xml'
      STYLE=analysis;

PROC REPORT DATA=Staffing NOWD;
      COLUMNS Unit Num_Emp Total_Sal Avg_Sal;
      DEFINE Avg_Sal / STYLE={tagattr="formula:RC[-1]/RC[-2]"};
RUN;

ODS tagsets.ExcelXP CLOSE;
```

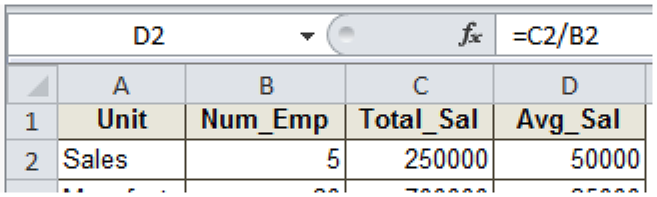
The output in Excel looks the same on the surface.



	A	B	C	D
1	Unit	Num_Emp	Total_Sal	Avg_Sal
2	Sales	5	250000	50000
3	Manufact	20	700000	35000
4	Engineer	7	560000	80000
5	Support	8	600000	75000

Display 3. Excel export #2

But looking again at the average salary field in cell D2 this time we see that cell contains the actual formula for the average salary calculation and not just the value. So now if the number of employees or total salary were changed the average salary would automatically recalculate in Excel. This same method can also be used for many other formulas and functions in Excel.



D2 =C2/B2

	A	B	C	D
1	Unit	Num_Emp	Total_Sal	Avg_Sal
2	Sales	5	250000	50000

Display 4. Excel export #2 detail

This same method can be used in PROC PRINT to produce the same output. In PROC PRINT you would use a VAR statement along with the style option to define the variable to export a formula.

```
PROC PRINT data=Staffing noobs;
       var Unit Num_Emp Total_Sal;
       var Avg_Sal / style={tagattr="formula:RC[-1]/RC[-2]"};

run;
```

R1C1 REFERENCE STYLE

You must use the R1C1 reference style in any formula exported from SAS using the ExcelXP tagset, using the more familiar A1 style with will not work. This reference style refers to the row R and column C number so R1C1 is the same as A1. If no number is put after the R or C then it will use the row or column started in. Using [ ] around the number makes the reference relative, if not it is absolute (similar to using $), for example if you place a formula into cell B2, the relative reference referring to cell A1 would be R[-1]C[-1] or the absolute reference referring to cell C3 would be R[1]C[1]. Another example is a formula using R1C1 reference style SUM(R[-5]C:R[-1]C) would be the sum of the 4 cells above the formula cell. I find that it is often helpful to export the report to Excel without formulas, lay out my formulas in the Excel report, and then go back to build the formulas in SAS.

MISSING VALUES

The ExcelXP tagset will only export a formula into the cell in Excel if the observation in SAS for the chosen variable has a non-missing value, for example 0 not “.” This can be used to create a dummy variable in a data set that will be filled with formulas when exported so you can control which cells receive a formula. For example you could place any non-missing value where you want a formula and a missing values for the where you don’t want a formula.

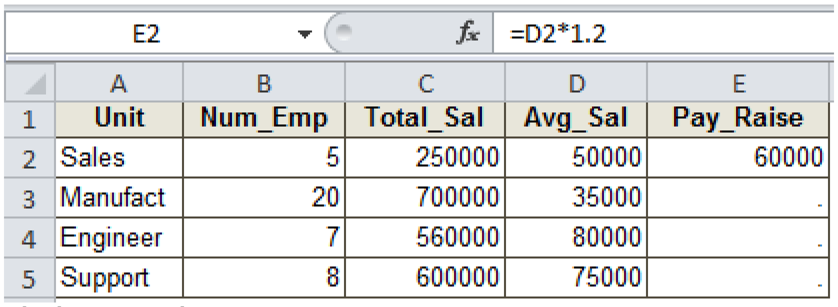
This method of using missing values could be applied if I wanted to know what the average salary would be after a 20% pay raise but only for sales personnel. I can build a new variable (Pay_Raise) in SAS but instead of performing the calculation in SAS I will use an IF/THEN statement to put a non-missing value in the new variable for the Sales observation and leave the rest of the observations as missing.

```
        IF Unit = 'Sales' THEN Pay_Raise = 0;
```

I will then DEFINE the new variable as part of the PROC REPORT and use the TAGATTR= style attribute to export the formula used to calculate the average salary after a 20% pay raise.

```
        DEFINE Pay_Raise / STYLE={tagattr="formula:RC[-1]*1.2"};
```

Now looking at the output in Excel you see that the formula was only exported for the Sales unit which had a non-missing value but the remaining units which had missing values for Pay_Raise did not receive the formula.



E2 =D2*1.2

	A	B	C	D	E
1	**Unit**	**Num_Emp**	**Total_Sal**	**Avg_Sal**	**Pay_Raise**
2	Sales	5	250000	50000	60000
3	Manufact	20	700000	35000	.
4	Engineer	7	560000	80000	.
5	Support	8	600000	75000	.

Display 5. Excel export #3

CONCLUSION

In a work environment requiring that Excel tables have formulas and calculations that automatically update. You can easily meet this requirement using SAS without resorting to updating Excel files manually or utilizing Excel Macros by using the ExcelXP tagset.

ACKNOWLEDGEMENTS

The assistance provided by Elizabeth Schreiber for both this paper and my journey of learning SAS has been greatly appreciated.

CONTACT INFORMATION

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

Joseph Skopic
jskopic@gmail.com