Printable Spreadsheets Made Easy: Utilizing the SAS® Excel XP Tagset

Rick Andrews, UnitedHealth Group, Cary, NC

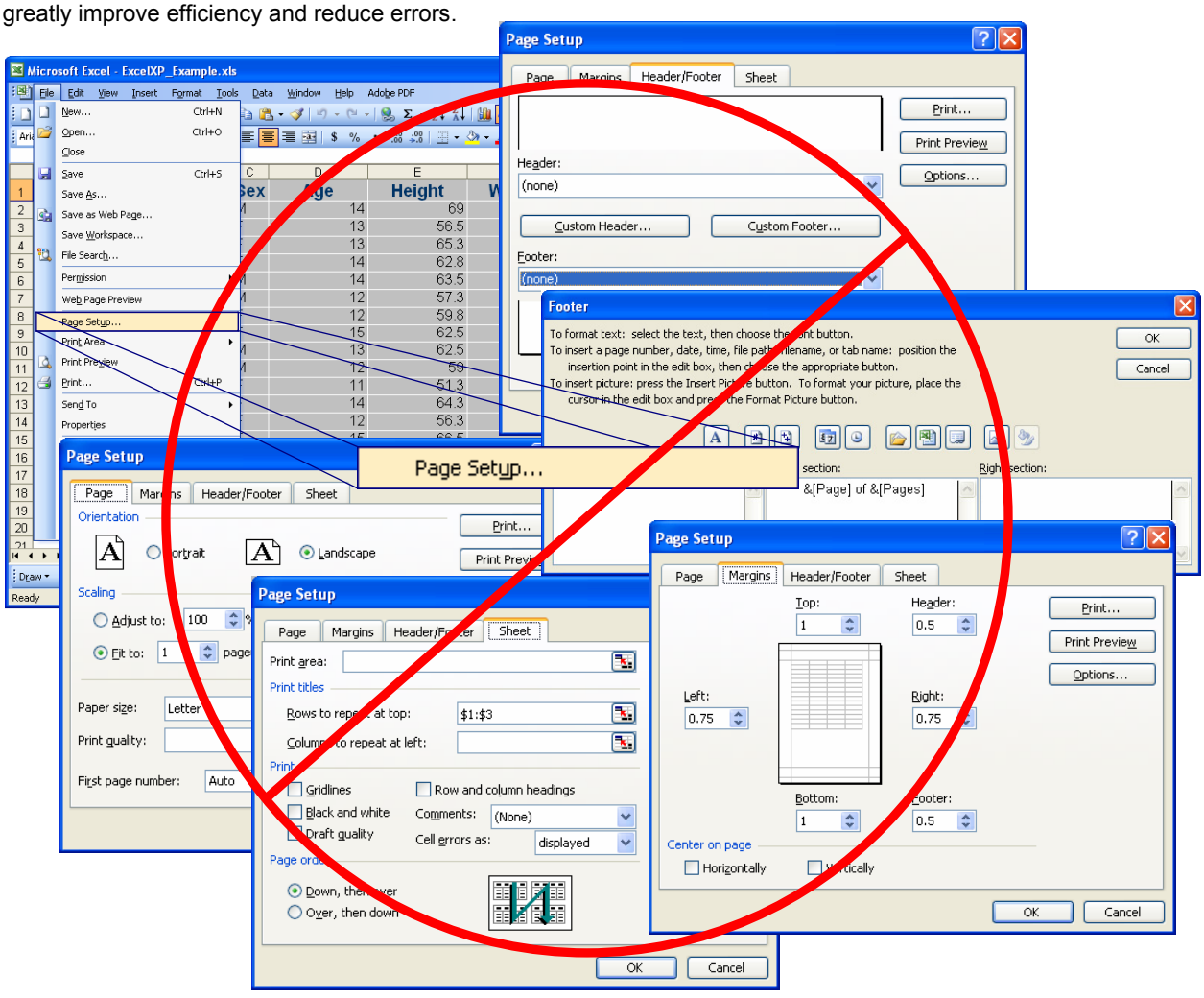
ABSTRACT

The SAS System offers myriad techniques for reporting on data within Microsoft® Excel. Depending on the task at hand SAS Access® or the Output Delivery System (ODS) might be good choices; Dynamic Data Exchange (DDE) or the old standby, Comma-Separated Values (CSV). This paper describes a method of creating multi-tab, print-ready reports using the Excel XP tagset available in version 9.1. This feature of Base SAS can greatly minimize the manual and repetitious task of preparing headers, footers, and various other formatting needs.

INTRODUCTION

The tagset utilizes the eXtended Markup Language (XML); an open standard for the definition, transmission, validation, and interpretation of data. The standard was developed by the Worldwide Web Consortium (W3C) to provide an efficient way to manage self-documenting data files (Gebhart, 2008). Knowledge of XML is not required to use the Excel XP tagset. The SAS code necessary is very similar to most other ODS processes and only a handful of options are needed to create spreadsheets ready for publication.

The advantage of using the tagset is the reduction in formatting time. Configuring one spreadsheet may take only a few moments, though if the same spreadsheet needs changing repeatedly, or if a similar document needs creating for multiple iterations, the formatting can be come very cumbersome. Knowing only a few of new Excel XP options can greatly improve efficiency and reduce errors.



REQUIREMENTS

The techniques presented in this paper utilize technologies implemented in Base SAS 9.1 or later, on any supported operating system and hardware, and Microsoft® Excel 2002 or later. The current version of the ExcelXP tagset has undergone various revisions since initial release. The latest tagset should be downloaded from the SAS Research & Development (R&D) website located at: http://support.sas.com/rnd/base/ods/odsmarkup

Notice the current version of the tagset at the writing of this paper is 1.86; see Figure 1, ExcelXP Tagset Download. Also located on this site are links to various examples, demos, and tutorials to give even the most novice individual a head-start with the exciting new world of markup capabilities.

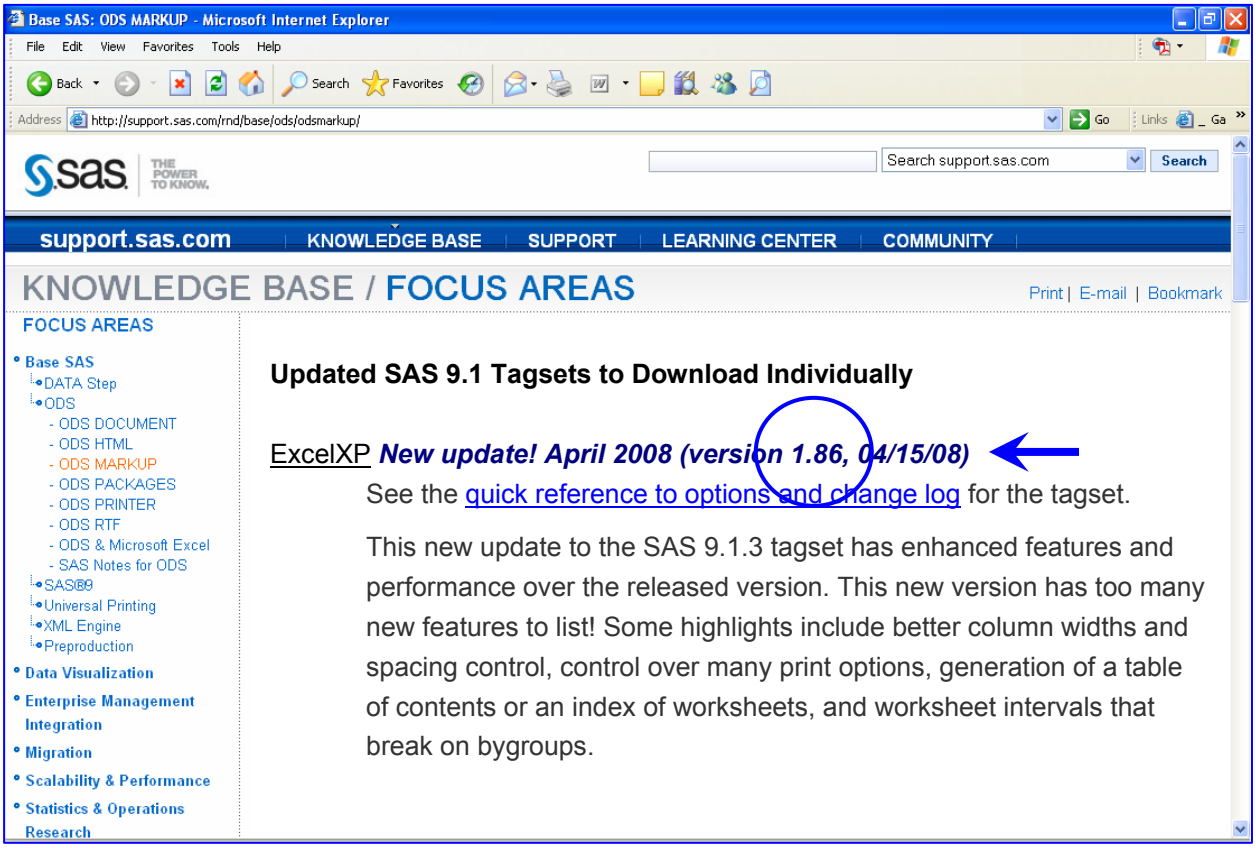


Figure 1: ExcelXP Tagset Download

UPDATE TAGSET

The ExcelXP tagest is created using the Template procedure, though absolutely no knowledge of the PROC is required. Copy and paste the code located at the website listed in Figure 1 into a SAS program editor and submit it. This will create or update the current version in the SASHELP.TMPLMST catalog. The history and usage of the tagset exist directly within the code and the `log_note`, shown below, indicates the version and when it was updated.

```
proc template;
...
  define Tagset Tagsets.ExcelXP;
       parent = Tagsets.ExcelBase;
  end;
...
  log_note = "NOTE: This is the Excel XP tagset (SAS 9.1.3, v1.86, 04/15/08)
...
run;
```

COMPARISON

Creating Excel spreadsheets from SAS has been available since, well, forever. If only rows and columns of data are required a CSV file may suffice and there is no coding required as a handy export wizard to guide through the process. If data need formatting in a special way DDE or ODS HTML can be used, though trying to compare DDE, ODS HTML, and the Excel XP tagset might be related to apples and oranges (or comparing Excel XP apples from a tree growing in the backyard with DDE or ODS HTML oranges grown in South America, where instead of picking the fruit from the branch, they are shipped via boat, train, and truck to the grocery store and are still not available until purchased).

The end result is Excel, but the difference is HOW to get there. In this case, the journey is everything.

DDE

DDE technology uses the Excel 4 Macro (XLM) language, which is a predecessor to Visual Basic for Applications (VBA), Open Database Connectivity (ODBC) and Object Linking and Embedding (OLE).

PRO - Currently the only method that can be used to perform very specific updating to individual cells within a new or existing spreadsheet

CON - Both SAS and Excel need to be open for DDE commands to populate the Excel spreadsheet and the computer is tied up during processing

ODS HTML

Opens, manages, or closes the Hypertext Markup Language (HTML) destination, which produces HTML 4.0 output that contains embedded stylesheets. Changing the HTML extension to XLS opens the page in Excel.

PRO - Allows the inclusion of SAS/GRAPH® images, only requires the Base SAS product, and can be created on any operating system

CON - Cannot use this technique to update a single cell in an existing spreadsheet and there is not an OPTIONS statement used in the destination

EXCEL XP TAGSET

Microsoft announced the Spreadsheet version of the eXtended Markup Language (XML) in Office 10 (Office 2000), which allows the definition of elements based on the needs of the task in lieu of hard-coded within the software.

PRO - Can execute in batch mode from SAS on any supported platform or operating system and Excel need not be open for this method to work

CON - Cannot use this technique to update a single cell in an existing spreadsheet and does not support insertion of graphic images

The PROs and CONs mentioned here are not all-inclusive and advances in newer versions of Base SAS may have changed those presented. They are given as examples to demonstrate that one option may provide greater flexibility while another ease of use. The choice to use one method over the other truly depends on the situation and the task at hand. Quite often an outcome might require a pre-formatted spreadsheet populated with very specific attributes and the Excel XP tagset may not be the right choice; DDE could be the only alternative.

The Excel XP tagset is a new technology and is not found in the SAS OnLineDoc®, as of the writing of this paper, though much documentation has been provided directly within the tagset itself. Also, as the expertise grows ever more SAS papers are being written on the topic. The time spent learning this new tool should be greatly outweighed by the time savings of formatting repetitive reports in Excel.

USAGE

In its simplest form, the Excel XP tagset is used very much like the ODS HTML destination. Figure 2: Excel XP Tagset vs. ODS HTML shows the differences in the syntax are the opening and closing of destinations. The output is slightly different as the default STYLE for each is uncommon and the name of the tab created using Excel XP is "Table 1 - Data Set SASHELP.CLAS" denoting the SAS data set that was used versus ODS HTML, which created a tab called "HTML_Example", the name of the workbook itself.

```
ODS TAGSETS.EXCELXP
    FILE='C:\ExcelXP_Example.xls';

  PROC PRINT DATA=sashelp.class; RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

```
ODS HTML
    FILE='C:\HTML_Example.xls';

  PROC PRINT DATA=sashelp.class; RUN;

ODS HTML CLOSE;
```

Microsoft Excel - ExcelXP_Example.xls

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84
10	John	M	12	59	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90
13	Louise	F	12	56.3	77
14	Mary	F	15	66.5	112
15	Philip	M	16	72	150
16	Robert	M	12	64.8	128
17	Ronald	M	15	67	133
18	Thomas	M	11	57.5	85
19	William	M	15	66.5	112

Table 1 - Data Set SASHELP.CLAS

Microsoft Excel - HTML_Example.xls

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5
2	Alice	F	13	56.5	84
3	Barbara	F	13	65.3	98
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84
10	John	M	12	59	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90
13	Louise	F	12	56.3	77
14	Mary	F	15	66.5	112
15	Philip	M	16	72	150
16	Robert	M	12	64.8	128
17	Ronald	M	15	67	133
18	Thomas	M	11	57.5	85
19	William	M	15	66.5	112

HTML_Example

Figure 2: Excel XP Tagset vs. ODS HTML

This is where much of the similarity ends. Each can accept common ODS options such as STYLE=minimal for example, though only the Excel XP tagset has an **OPTIONS** statement containing over fifty elements used in formatting the spreadsheet. One of these options is called Doc, which tells SAS to output the available help to the Log. Notice in Figure 3: ODS OPTIONS vs. Excel XP OPTIONS, the ODS STYLE option is used exactly as it would in ODS HTML, though an OPTIONS statement has been added that contains the Doc option set to the 'Help' parameter surrounded by parentheses.

```
ODS TAGSETS.EXCELXP

FILE='C:\NESUG\ExcelXP_Example2.xls'
    STYLE = minimal
    OPTIONS ( Doc = 'Help' );

  PROC PRINT DATA=sashelp.class; RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

Figure 3: ODS OPTIONS vs. Excel XP OPTIONS

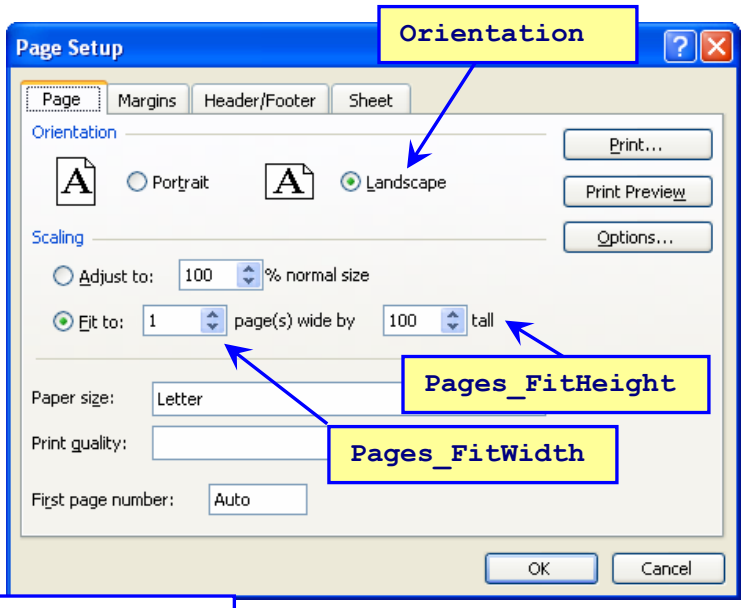
Below is the Help output created for the Doc option obtained from the SAS Log:

```
Doc:  No default value.
  Help: Displays introductory text and
        available options in full detail.
  Quick: Displays introductory text and
         an alphabetical list of options,
         their current value, and short
         description.
  Settings: Displays config/debug settings.
  Changelog: Lists the changes in reverse chronological order.
  All: Shows the output from all the help options.
```

PAGE SETUP

Getting familiar with the fiftly-plus available options can be a daunting task. This paper describes only some of the possibilities and how they relate to the Excel spreadsheet. One of the great nuances of the Excel XP tagset is the ability to easily setup the printing options.

As shown in Figure 4: Page Setup Options, the options for setting the Orientation and Scaling are being displayed. This is a rather easy task to perform manually if only one file were being created, though if many files need to formatted, the task could become rather cumbersome and prone to error. The code below signifies how to perform these options using the tagset.




```
ODS TAGSETS.EXCELXP
    FILE='C:\NESUG\ExcelXP_Example3.xls'
    STYLE=minimal
    OPTIONS ( Orientation      = 'landscape'
              FitToPage        = 'yes'
              Pages_FitWidth   = '1'
              Pages_FitHeight  = '100' );

PROC PRINT DATA=sashelp.class; RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

Figure 4: Page Setup Options

Notice the Doc = 'Help' option was removed and replaced with Orientation, FitToPage, Pages_FitWidth, and the Pages_FitHeight. Some of the options like Orientation only have two possible parameters; landscape or portrait, while others like the Pages_Fit_Height option are determined by necessity.

Also note the parameters and their values are separated by an equal sign, surrounded with quotes, and all options surrounded with parentheses. The semi-colon appears at the end of the entire statement.

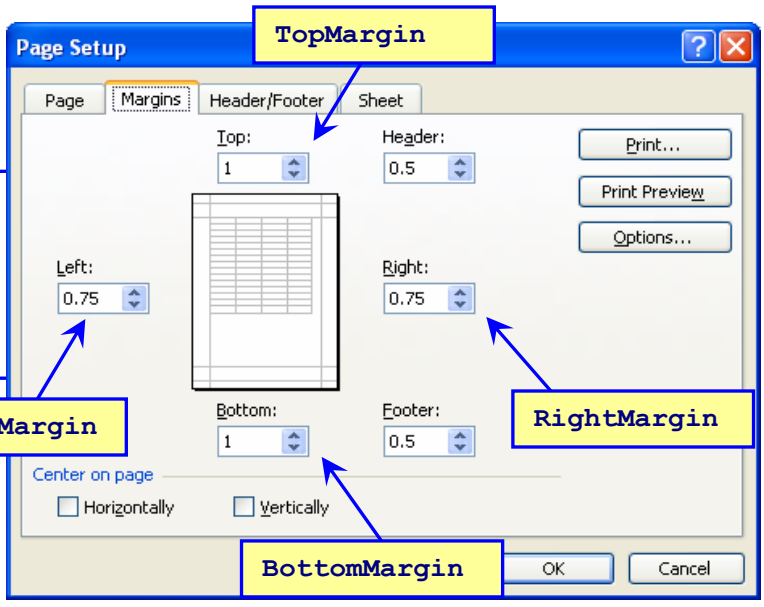
MARGINS

The margins of the printable area of the spreadsheet are set using a SAS OPTIONS statement in lieu of an Excel XP option, as shown in Figure 5: Margins.

```
OPTIONS LeftMargin   = .5in
        RightMargin  = .5in
        TopMargin    = .5in
        BottomMargin = .5in ;
```

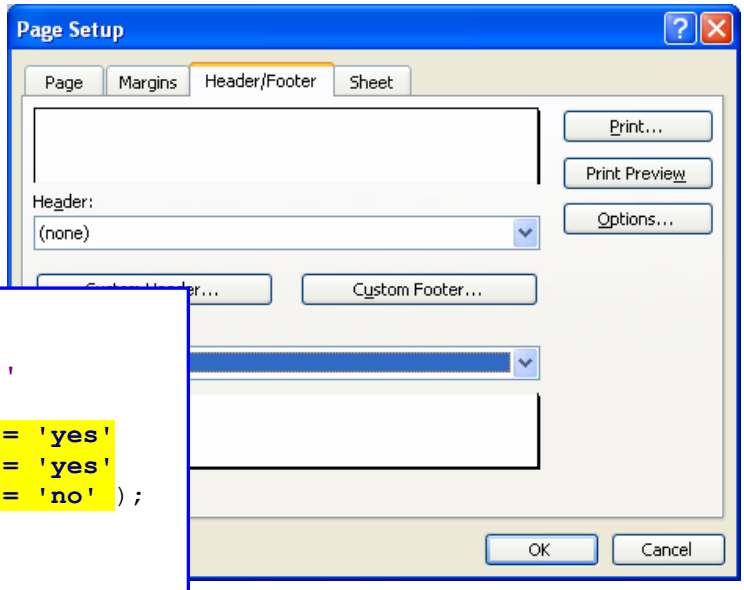
Figure 5: Margins

The Header and Footer margins can be set using the Print_Header_Margin and Print_Footer_Margin parameters of the Excel XP tagset.



HEADERS & FOOTERS

The Headers and Footers can be set by the Title and Footnote statements respecively. Two Excel XP options exist to control whether a title or footnote is shown within the spreadsheet itself or in the printable section alone. The following code will create a file containing a title in the worksheet itself and a footnote only when printed.




```
ODS TAGSETS.EXCELXP
    FILE='C:\ExcelXP_Example4.xls'
    STYLE=minimal
    OPTIONS ( Center_Horizontal  = 'yes'
              Embedded_Titles    = 'yes'
              Embedded_Footnotes = 'no' );

TITLE1 'SAS is Great';
TITLE2 'SAS is Good';
FOOTNOTE1 'Let us Thank';
FOOTNOTE2 'Jim we Should';

PROC PRINT DATA=sashelp.class; RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

Figure 6: Titles & Footnotes

Notice there is no footer in the worksheet as indicated in Figure 7: Worksheet with NO Footer and there is indeed a footer in Figure 8: Print Preview with Footer. This is useful when a title within the worksheet itself is desired and a footnote only need exist in the footer. The Center_Horizontal option is also used to center the output.

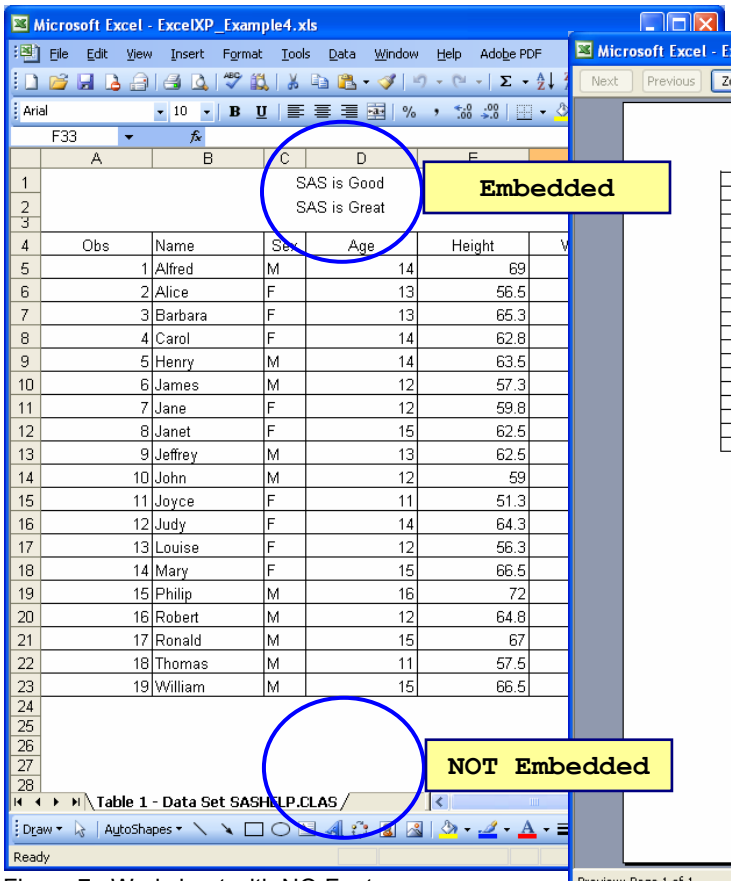


Figure 7: Worksheet with NO Footer

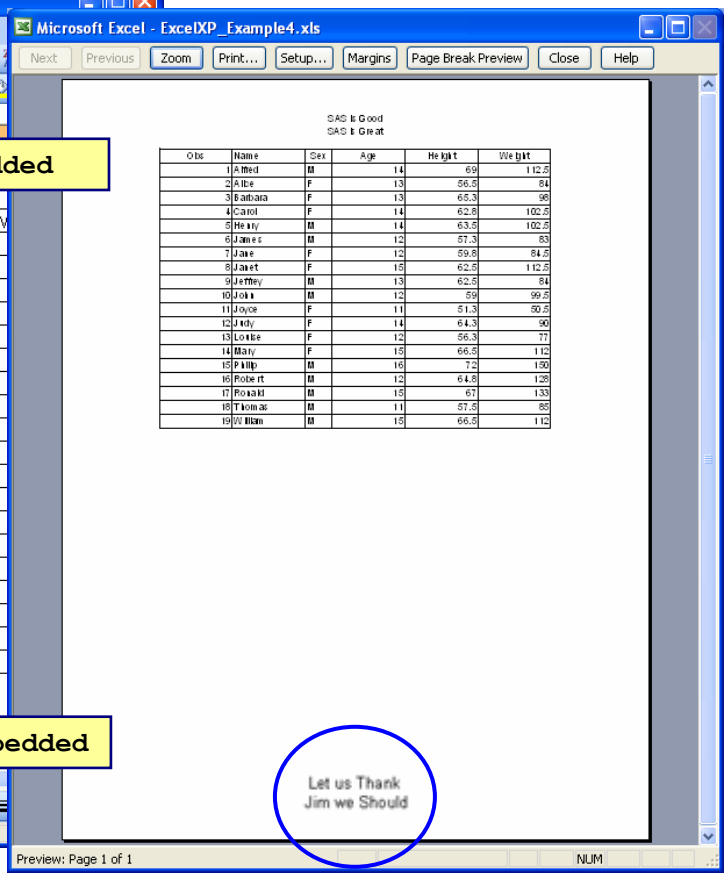


Figure 8: Print Preview with Footer

PAGE of PAGES

If **Embedded_Footnotes** are on, the **Print_Footer** option will be used as the footer for printing. Everything about the appearance of the footer can be controlled with this value. The easiest way to create a header or footer is to first create them in Excel, save the workbook as an XML Spreadsheet, open the saved file in Notepad, and search for <header or <footer. The exact syntax found within can then be used in SAS.

Below are some of the options available as listed within the Doc = 'Help' option of the Excel XP tagset:

Newline:
Page Number:	&P
Pages:	&N
Date:	&D
Time:	&T
File Path:	&Z&
File:	&F
Sheet Name:	&A
Underline:	&U
Font Size:	&8

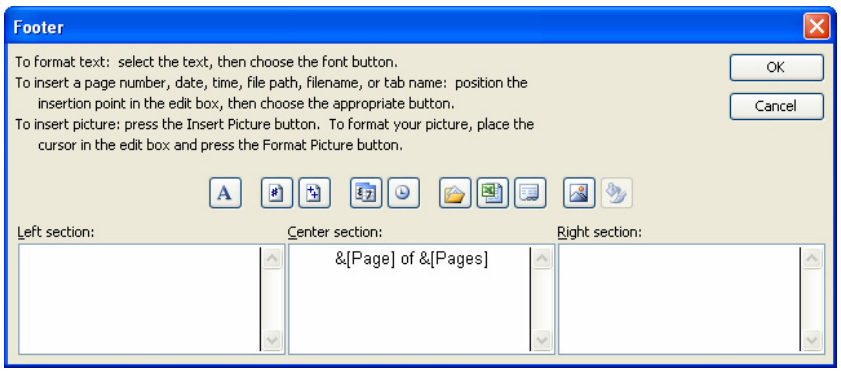


Figure 9: Page of Pages Window

Other options include changing the font and other characteristics such as bold and italic.

Many possibilities exist to create almost any header or footnote that can be imagined. Though watch out for spaces between the "&" as this can cause unexpected results. It might be best to create the desired result in Excel first and view the XML syntax as described above.

```
TITLE; FOOTNOTE;

ODS TAGSETS.EXCELXP
    FILE='C:\NESUG\ExcelXP_Example5.xls'
    STYLE=minimal
    OPTIONS ( Center_Horizontal  = 'yes'
              Embedded_Titles    = 'yes'
              Embedded_Footnotes = 'yes'
              Print_Footer       = 'Let us Thank
&#13; Jim we Should &#13; Page: &amp;P of Pages:
&amp;N' );

TITLE1 'SAS is Great';
TITLE2 'SAS is Good';

PROC PRINT DATA=sashelp.class; RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

FOOTNOTEs moved to **Print_Footer**

FOOTNOTE

Notice in Figure 10: Page of Pages Preview the FOOTNOTE statements have been removed. If they had not the footnotes would appear twice; once at the end of each report and another at the end of each page.

This can be a bit confusing at first. To completely understand the result, create a report with and without both the FOOTNOTE statement and the Print_Footer option to get a feel of the correct syntax.

The Print_Footer shown is actually a long text string containing the footnote verbiage and special XML characters, shown above to control the formatting of the footnote.

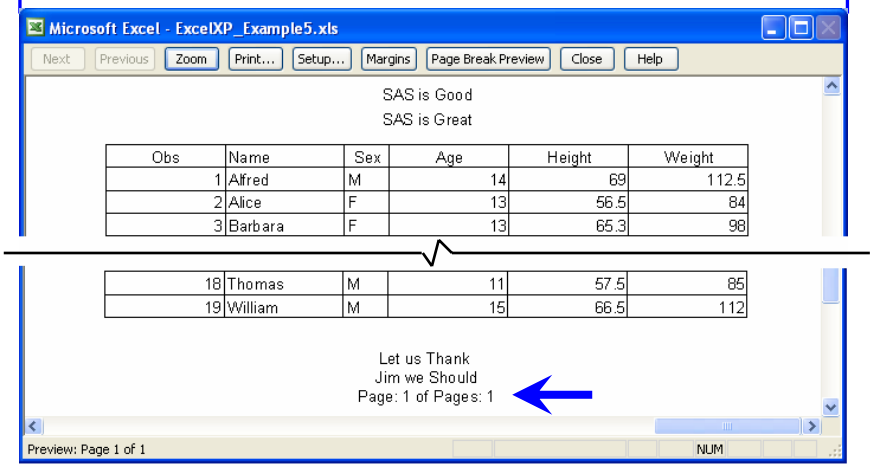


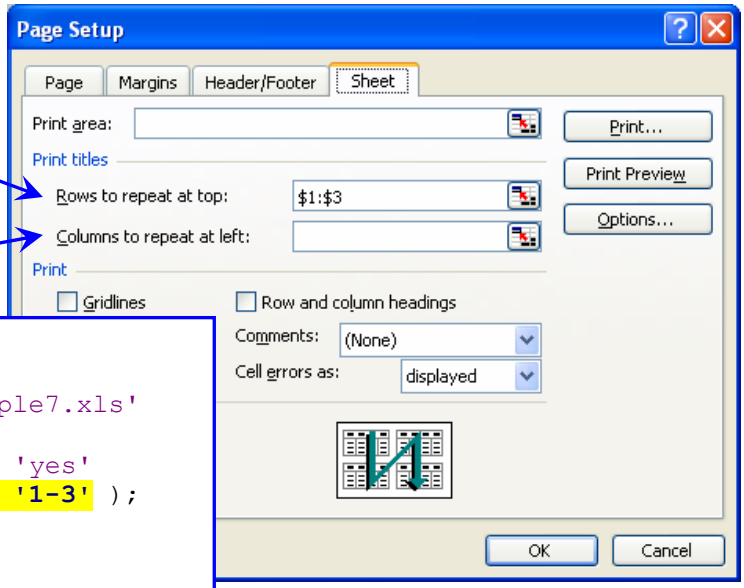
Figure 10: Page of Pages Preview

Example:

```
Print_Footer = 'Let us Thank &#13; Jim we Should &#13; Page: &amp;P of Pages: &amp;N'
```

ROW REPEAT

Another useful feature of the Excel XP tagset is the Row Repeat option. This will identify the rows to be repeated when the worksheet is printed. Syntax for this option is listed inside of Figure 11: Row Repeat.




```
ODS TAGSETS.EXCELXP
    FILE='C:\NESUG\ExcelXP_Example7.xls'
    STYLE=minimal
    OPTIONS ( Embedded_Titles = 'yes'
              Row_Repeat      = '1-3' );

TITLE 'Row Repeat Option';

PROC PRINT DATA=sashelp.air; RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

Figure 11: Row Repeat

MULTIPLE WORKSHEETS

One of the most useful aspects of the tagset is the ability to create multiple tabs or worksheets within the same workbook. In this example there are two worksheets being created and given the names of Shoes and Class respectively. This is accomplished using the Sheet_Name option.

The gist of this option is to open the Excel XP destination, then using the Sheet_Name option give the first report a name. Then before closing the destination provide SAS with a second destination and supply another sheet name.

This can be done for as many reports as are required for the project. Once all tabs have been created the ODS destination is closed and the spreadsheet is created and saved by SAS. Notice the FILE and STYLE options of the ODS statement are not repeated. They only need to be set once. This is also true of the OPTIONS of the Excel XP tagset as will be demonstrated later.

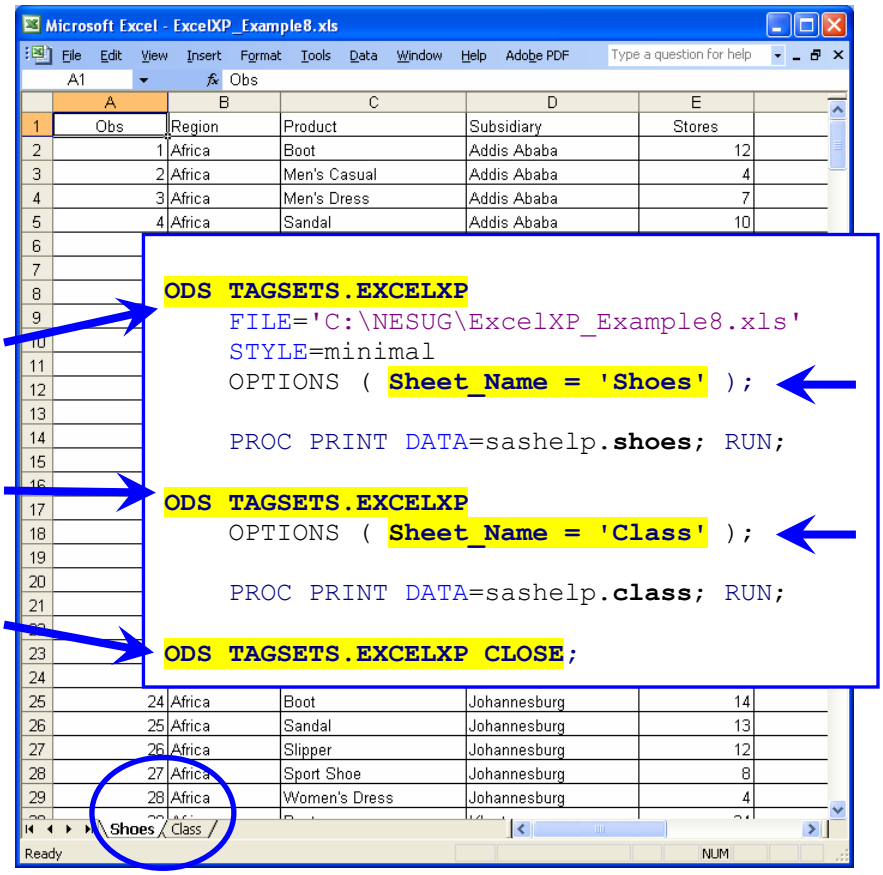


Figure 12: Multiple Worksheets

SHEET INTERVAL

If the desired result is to have both reports within the same worksheet the Sheet_Interval is used without reopening the Excel XP destination. This can be helpful when reports are small and can fit onto one page.

Other parameters for the Sheet_ Interval option includes Table, Page, Bygroup, and Proc, see the Doc = 'Help' for more information.

```
ODS TAGSETS.EXCELXP
    FILE='C:\NESUG\ExcelXP_Example9.xls'
    STYLE=minimal
    OPTIONS( Sheet_Name      = 'Shoes & Class'
             Sheet_Interval = 'none');

PROC PRINT DATA=sashelp.shoes; RUN;
PROC PRINT DATA=sashelp.class; RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

Figure 13: Two Reports on Same Worksheet

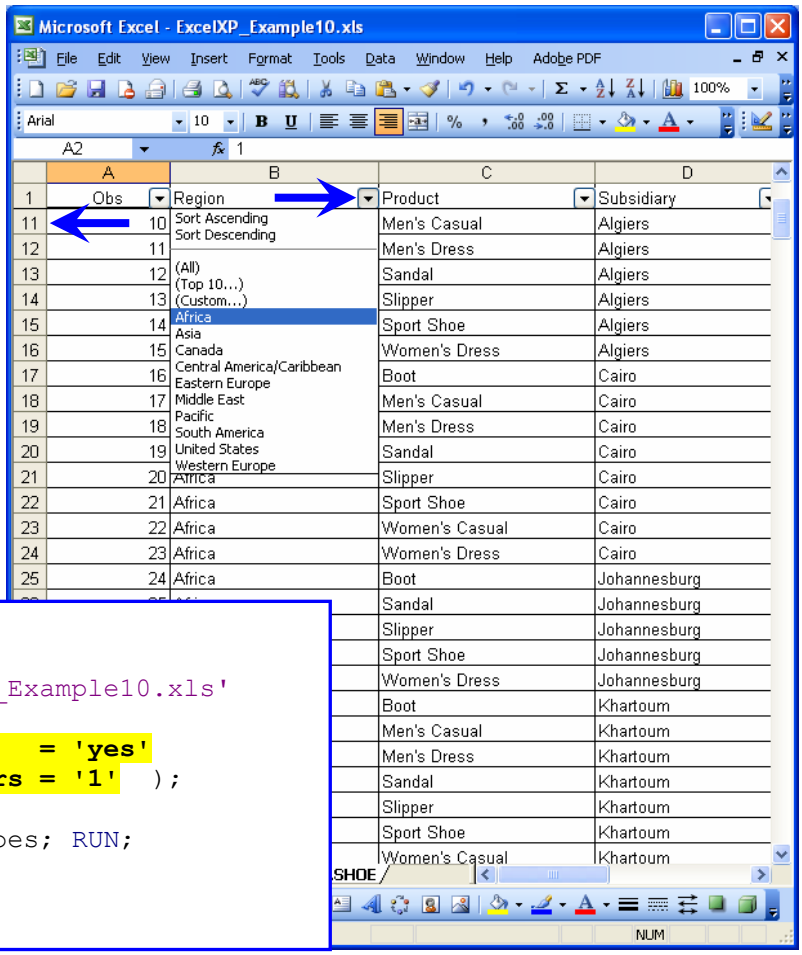
FORMATTING & READABILITY

Up to this point in the document focus has been on **printability** of the Excel spreadsheet, though many options exist to incorporate a host of various formatting and **readability** options available to the Excel XP tagset. The following will describe mechanisms to allow for reports to become boardroom ready at the press of a SAS button.

AUTOFILTER & FROZEN HEADERS

The Autofilter and the Frozen_Header options make reading and filtering data extremely helpful, only take two lines of code, and will probably take less time to write than opening the spreadsheet.

Notice in the Figure 14: Auto Filter and Frozen Headers the rows are about to be filtered to read only data from Africa and the Header row has been frozen to allow the scrolling of data. Since titles were not included the Frozen_Headers option was set to one ('1'). If titles had been inserted this value would change to account for the number of new rows introduced.




```
ODS TAGSETS.EXCELXP
    FILE='C:\NESUG\ExcelXP_Example10.xls'
    STYLE=minimal
    OPTIONS ( Autofilter      = 'yes'
              Frozen_Headers = '1'  );

PROC PRINT DATA=sashelp.shoes; RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

Figure 14: Auto Filter and Frozen Headers

COLUMN WIDTH

The Absolute_Column_Width parameter works similarly to the Default_Column_Width option only the absolute widths are used regardless of what may be provided by the procedure.

In the examples shown in the Figure 15: Column Widths and Row Heights, there are five reports being created depicting the outcome of using the Absolute_Column_ Width parameter with the PRINT and REPORT procedures.

1 PROC PRINT is being used with no options other than the Sheet_Name. This uses the default value of the Absolute_Column_Width of 'none' to be used, which lets the procedure identify the width of the columns.

2 The Absolute_Column_Width can be a single value, which will set all widths to the same setting or it can be a list of numbers separated by commas. The order of the values corresponds with the order of the variables on the report. Notice the Sex variable in example 2 is the third one on the report, though it is not the third variable in the data set. The Obs field is included when using PROC PRINT.

3 Example 3 uses the NOOBS option to suppress the Obs variable from being printed on the report. The Absolute_Column_ Width is the same as that used in example 2 because the Excel XP tagset uses the values of the previous. Note the Age variable is now set to a length of 10 in lieu of Sex.

4 Example 4 sets the Absolute_Column_Width to a value of 'NONE', which sets it back the default value and allows the widths to be determined by the procedure. In this example a PROC REPORT is being used. Notice the widths are all set to the same value unlike those in example 1, which is the default for a PROC PRINT.

5 This example sets the Absolute_Column_Width to that which is used in examples 2 and 3. Note the outcome is the same in that the third variable has a length set to 10.

ROW HEIGHT

Also shown in example 5 is the Autofit_Height option, which set the height of the row to best fit the height of the point size being used for the data values.

```
ODS TAGSETS.EXCELXP
    FILE='C:\NESUG\ExcelXP_Example11.xls'
    STYLE=minimal
    OPTIONS ( Sheet_Name = 'Print-Default' );
  PROC PRINT DATA=sashelp.class(OBS=2);
```

1

Microsoft Excel - ExcelXP_Example11.xls

	A	B	C	D	E	F
1	Obs	Name	Sex	Age	Height	Weight
2	1	Alfred	M	14	69	112.5
3	2	Alice	F	13	56.5	84
4						

Print-Default | Print-OBS | Print-NOOBS | Report-Default | Report-Options

```
ODS TAGSETS.EXCELXP
    OPTIONS ( Sheet_Name = 'Print-OBS'
    Absolute_Column_Width = '5,5,10,5,5' );
  PROC PRINT DATA=sashelp.class(OBS=2);
```

2

Microsoft Excel - ExcelXP_Example11.xls

	A	B	C	D	E	F	G	H
1	Obs	Name	Sex	Age	Height	Weight		
2	1	Alfred	M	14	69	112.5		
3	2	Alice	F	13	56.5	84		
4								

Print-Default | Print-OBS | Print-NOOBS | Report-Default | Report-Options

```
ODS TAGSETS.EXCELXP
    OPTIONS ( Sheet_Name = 'Print-NOOBS' );
  PROC PRINT DATA=sashelp.class(OBS=2) NOOBS;
```

Microsoft Excel - ExcelXP_Example11.xls

	A	B	C	D	E	F	G	H
1	Name	Sex	Age	Height	Weight			
2	Alfred	M	14	69	112.5			
3	Alice	F	13	56.5	84			
4								

3

Print-Default | Print-OBS | Print-NOOBS | Report-Default | Report-Options

```
ODS TAGSETS.EXCELXP
    OPTIONS ( Sheet_Name = 'Report-Default'
    Absolute_Column_Width = 'NONE' );
  PROC REPORT DATA=sashelp.class(OBS=2) NOWD;
```

4

Microsoft Excel - ExcelXP_Example11.xls

	A	B	C	D	E	F	G	H
1	Name	Sex	Age	Height	Weight			
2	Alfred	M	14	69	112.5			
3	Alice	F	13	56.5	84			
4								

Print-Default | Print-OBS | Print-NOOBS | Report-Default | Report-Options

```
ODS TAGSETS.EXCELXP
    OPTIONS ( Sheet_Name = 'Report-Options'
    Absolute_Column_Width = '5,5,10,5,5'
    Autofit_Height = 'YES');
PROC REPORT DATA=sashelp.class(OBS=2) NOWD; RUN;
```

5

Microsoft Excel - ExcelXP_Example11.xls

	A	B	C	D	E	F	G	H
1	Name	Sex	Age	Height	Weight			
2	Alfred	M	14	69	112.5			
3	Alice	F	13	56.5	84			
4								

Print-Default | Print-OBS | Print-NOOBS | Report-Default | Report-Options

```
ODS TAGSETS.EXCELXP CLOSE;
```

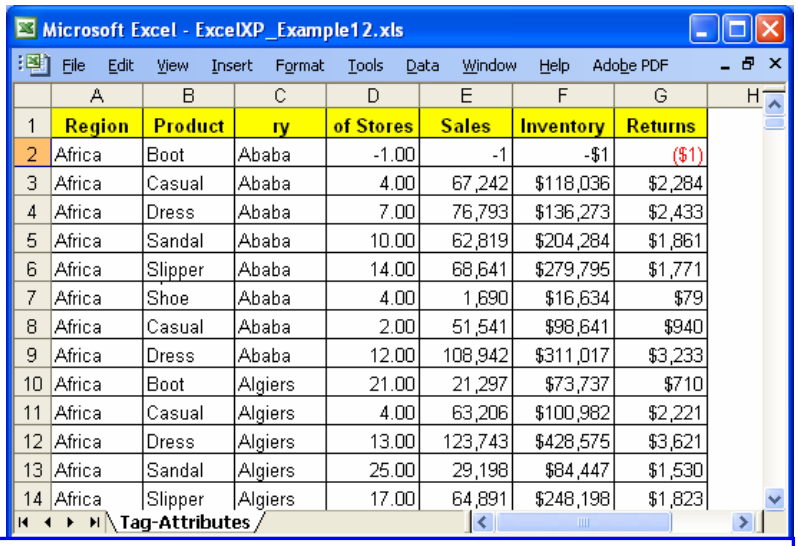
Figure 15: Column Widths and Row Heights

TAG ATTRIBUTES

The TAGATTR parameter can be used within a PROC REPORT to display the variables to an Excel supplied format. Notice row 2, columns D through F has been set to a negative 1. The formats supplied in the TAGATTR in Figure 16: Tag Attribute Examples are set to the following values:

```
0.00
#,##0
$#,##0
$#,##0_);[Red]($#,##0)
```

These values are entered exactly as listed within the Format Cells window of Microsoft Excel as shown below.



	A	B	C	D	E	F	G
1	Region	Product	ry	of Stores	Sales	Inventory	Returns
2	Africa	Boot	Ababa	-1.00	-1	-$1	($1)
3	Africa	Casual	Ababa	4.00	67,242	$118,036	$2,284
4	Africa	Dress	Ababa	7.00	76,793	$136,273	$2,433
5	Africa	Sandal	Ababa	10.00	62,819	$204,284	$1,861
6	Africa	Slipper	Ababa	14.00	68,641	$279,795	$1,771
7	Africa	Shoe	Ababa	4.00	1,690	$16,634	$79
8	Africa	Casual	Ababa	2.00	51,541	$98,641	$940
9	Africa	Dress	Ababa	12.00	108,942	$311,017	$3,233
10	Africa	Boot	Algiers	21.00	21,297	$73,737	$710
11	Africa	Casual	Algiers	4.00	63,206	$100,982	$2,221
12	Africa	Dress	Algiers	13.00	123,743	$428,575	$3,621
13	Africa	Sandal	Algiers	25.00	29,198	$84,447	$1,530
14	Africa	Slipper	Algiers	17.00	64,891	$248,198	$1,823

```
ODS TAGSETS.EXCELXP
    FILE='C:\NESUG\ExcelXP_Example12.xls'
    STYLE=minimal
    OPTIONS ( Sheet_Name = 'Tag-Attributes' );

PROC REPORT DATA=sashelp.shoes NOWD
  STYLE(header)=[ BACKGROUND  = yellow
                  FONT_WEIGHT = bold ];

  COLUMN Region Product Subsidiary Stores Sales Inventory Returns;

  DEFINE Stores    / STYLE(column)={TAGATTR='format:0.00'};
  DEFINE Sales     / STYLE(column)={TAGATTR='format:#,##0'};
  DEFINE Inventory / STYLE(column)={TAGATTR='format:$#,##0'};
  DEFINE Returns   / STYLE(column)={TAGATTR='format:$#,##0_);[Red]($#,##0)'};
RUN;

ODS TAGSETS.EXCELXP CLOSE;
```

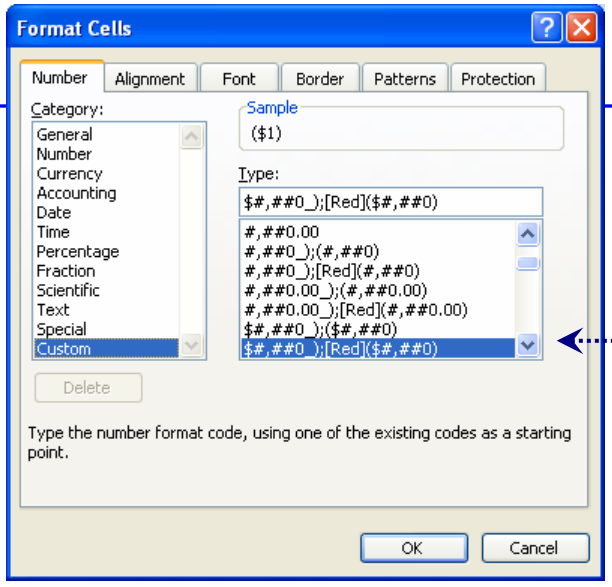


Figure 16: Tag Attribute Examples

```
ODS LISTING CLOSE;
ODS NORESULTS;

OPTIONS LeftMargin   = .5in
        RightMargin  = .5in
        TopMargin    = .5in
        BottomMargin = .5in;

ODS TAGSETS.EXCELXP
    FILE='C:\NESUG\ExcelXP_Example13.xls'
    STYLE=minimal
    OPTIONS ( Sheet_Name          = 'Shoes'
              Sheet_Interval      = 'proc'
              Orientation         = 'landscape'
              FitToPage           = 'yes'
              Pages_FitWidth      = '1'
              Pages_FitHeight     = '100'
              Center_Horizontal   = 'yes'
              Embedded_Titles     = 'no'
              Embedded_Footnotes  = 'no'
              Print_Header        = ''
              Print_Footer        = ''
              Autofilter          = 'yes'
              Frozen_Headers      = '1'
              Row_Repeat          = '1-3'
              Autofit_Height      = 'yes'
              Absolute_Column_Width = '6,10,8,8,8,8,8,8' );

TITLE1 'SAS is Good';
TITLE2 'SAS is Great';
FOOTNOTE1 'Let us Thank';
FOOTNOTE2 'Jim we Should';

PROC REPORT DATA=sashelp.shoes NOWD
  STYLE(header)=[ BACKGROUND  = yellow
                  FONT_WEIGHT = bold ];
  COLUMN Region Product Subsidiary Stores Sales Inventory Returns;
  DEFINE Stores    / STYLE(column)={TAGATTR='format:0.00' };
  DEFINE Sales     / STYLE(column)={TAGATTR='format:#,##0'};
  DEFINE Inventory / STYLE(column)={TAGATTR='format:$#,##0' };
  DEFINE Returns   / STYLE(column)={TAGATTR='format:$#,##0_);[Red]($#,##0)'};
RUN;

ODS TAGSETS.EXCELXP
    OPTIONS ( Sheet_Name   = 'Univariate-Stats'
              Print_Footer = 'Simple Footer'
              Absolute_Column_Width = '10' );

PROC UNIVARIATE DATA=sashelp.shoes;
  VAR Sales;
RUN;

ODS TAGSETS.EXCELXP CLOSE;
ODS LISTING;
ODS RESULTS;
```

Figure 17: Final Example

CONCLUSION

Many other options exist such as turning on the Auto Filter and Freezing Headers and Columns. Other options include, though are not limited to, adding formats to the data, creating drill-downs, and writing formulas in Excel. There will always be a need for the DDE, ODS HTML, and other methods of exporting data to Excel and now with the dynamic new ODS Excel XP tagset even more opportunities exist for create print-ready reports.

REFERENCES

Cisternas, Miriam and Ricardo Cisternas. “Reading and Writing XML files from SAS®,” SUGI 29 (2003), http://www2.sas.com/proceedings/sugi29/119-29.pdf.

DelGobbo, V. "A Beginner’s Guide to Incorporating SAS® Output in Microsoft Office Applications". SUGI (2003). http://www2.sas.com/proceedings/sugi28/052-28.pdf.

DelGobbo, V.. "Creating AND Importing Multi-Sheet Excel Workbooks the Easy Way with SAS® ". SUGI (2006), http://www2.sas.com/proceedings/sugi31/115-31.pdf.

Gebhart, Eric S. “ODS MARKUP: The SAS® Reports You've Always Dreamed Of,” SUGI (2004), http://www2.sas.com/proceedings/sugi30/085-30.pdf.

Gebhart, Eric S. “The Devil Is in the Details: Styles, Tips, and Tricks That Make Your Microsoft Excel Output Look Great!,”, SGF (2008), http://www2.sas.com/proceedings/forum2008/036-2008.pdf.

Microsoft Corporation. "XML Spreadsheet Reference", http://msdn2.microsoft.com/en-us/library/aa140066(office.10).aspx.

Pratter, Frederick. “Beyond HTML: Using the SAS System® Version 8.2 with XML,” SUGI 27 (2001), http://www2.sas.com/proceedings/sugi27/p002-27.pdf.

Parker, C. “Generating Custom Excel Spreadsheets using ODS”. *Proceedings of the Twenty-Eighth Annual SAS Users Group International Conference,* paper 12, 2003.

SAS Institute Inc. 2004. "Chapter 9: TEMPLATE Procedure: Creating a Style Definition". SAS 9.1 Output Delivery System, User's Guide. Cary, NC: SAS Institute Inc.

SAS Institute, “Technical Support Document #589b – Use PC-SAS 8.x and Higher to Read and Write Excel Files”. *http://ftp.sas.com/techsup/download/technote/ts325.pdf*, updated 2004.

Vyverman, K. “Using Dynamic Data Exchange to Pour SAS Data into Microsoft Excel”. *Proceedings of the Twenty-Sxith Annual SAS Users Group International Conference*, paper 11, 2001.

CONTACT INFORMATION

Your comments and questions are valued and encouraged.

Rick Andrews
UnitedHealth Group
101 Winstead Drive Suite 201
Cary, NC 27513
Rick.Andrews@Ingenix.com