Paper SAS1700-2015

Creating Multi-Sheet Microsoft Excel Workbooks with SAS®: The Basics and Beyond. Part 2

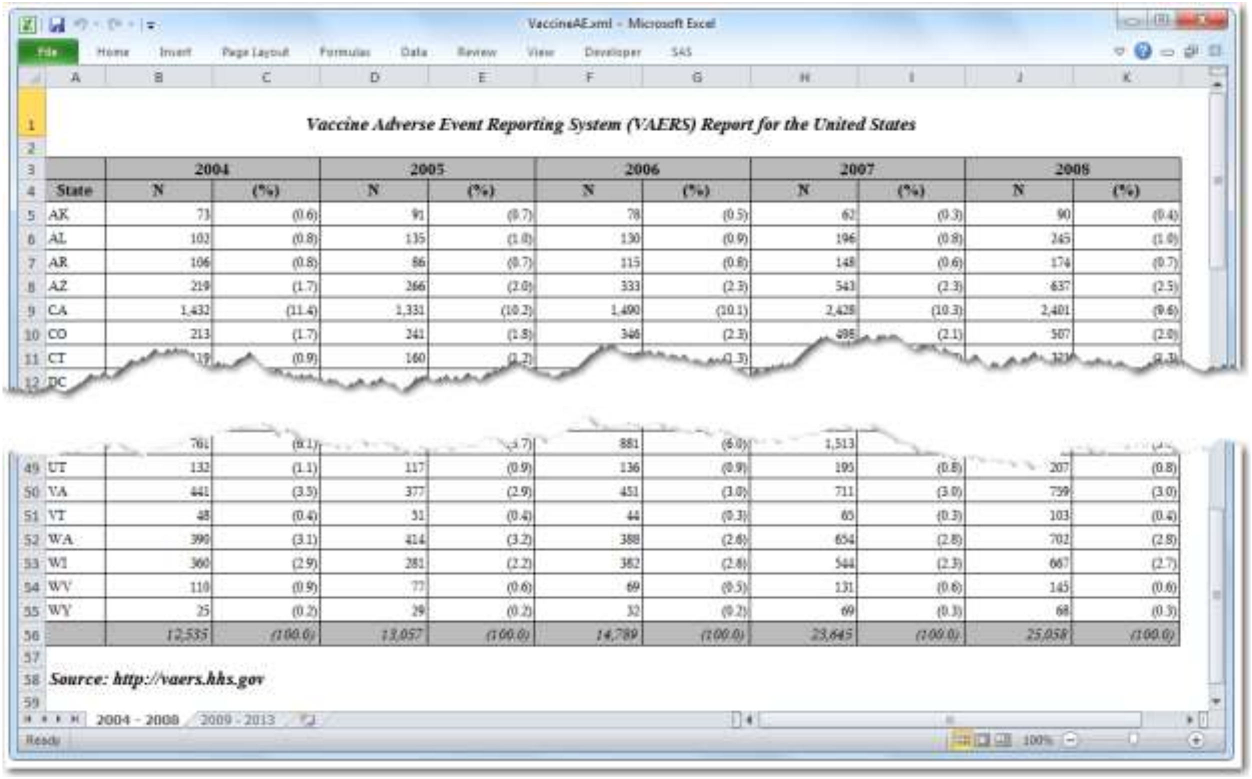
Vincent DelGobbo, SAS Institute Inc.

ABSTRACT

This presentation explains how to use Base SAS®9 software to create multi-sheet Excel workbooks. You learn step-by-step techniques for quickly and easily creating attractive multi-sheet Excel workbooks that contain your SAS® output using the ExcelXP Output Delivery System (ODS) tagset. The techniques can be used regardless of the platform on which SAS software is installed. You can even use them on a mainframe! Creating and delivering your workbooks on-demand and in real time using SAS server technology is discussed. Although the title is similar to previous presentations by this author, this presentation contains new and revised material not previously presented.

INTRODUCTION

This paper explains how to use Base SAS 9.1.3 or later to create the Excel workbook shown in Figure 1.



Vaccine Adverse Event Reporting System (VAERS) Report for the United States

	2004		2005		2006		2007		2008	
State	N	(%)	N	(%)	N	(%)	N	(%)	N	(%)
AK	73	(0.6)	91	(0.7)	78	(0.5)	62	(0.3)	90	(0.4)
AL	102	(0.8)	135	(1.0)	130	(0.9)	196	(0.8)	245	(1.0)
AR	106	(0.8)	86	(0.7)	115	(0.8)	148	(0.6)	174	(0.7)
AZ	219	(1.7)	266	(2.0)	333	(2.3)	543	(2.3)	637	(2.5)
CA	1,432	(11.4)	1,331	(10.2)	1,490	(10.1)	2,428	(10.3)	2,401	(9.6)
CO	213	(1.7)	241	(1.8)	346	(2.3)	498	(2.1)	507	(2.0)
CT	[illegible]	(0.9)	160	(2.2)	[illegible]	(2.3)	[illegible]	[illegible]	[illegible]	[illegible]
DC	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	761	(6.1)	[illegible]	[illegible]	881	(6.0)	1,513	[illegible]	[illegible]	[illegible]
UT	132	(1.1)	117	(0.9)	136	(0.9)	195	(0.8)	207	(0.8)
VA	441	(3.5)	377	(2.9)	451	(3.0)	711	(3.0)	759	(3.0)
VT	48	(0.4)	51	(0.4)	44	(0.3)	65	(0.3)	103	(0.4)
WA	390	(3.1)	414	(3.2)	388	(2.6)	654	(2.8)	702	(2.8)
WI	360	(2.9)	281	(2.2)	382	(2.6)	544	(2.3)	667	(2.7)
WV	110	(0.9)	77	(0.6)	69	(0.5)	131	(0.6)	145	(0.6)
WY	25	(0.2)	29	(0.2)	32	(0.2)	69	(0.3)	68	(0.3)
	12,535	*(100.0)*	*13,057*	*(100.0)*	*14,789*	*(100.0)*	*23,645*	*(100.0)*	*25,058*	*(100.0)*

Source: http://vaers.hhs.gov

Figure 1. Multi-Sheet Excel Workbook Generated by the ExcelXP ODS Tagset

The workbook includes two worksheets that contain vaccine adverse event data for the United States from 2004 to 2013.

Notable features of this workbook include the following:

1. Worksheet names are customized.
2. Title and footnote text are displayed in the document body.
3. Each worksheet prints on a single page.
4. The values in the numeric columns and summary row cells are displayed using Excel formats, not SAS formats.

The code in this paper was tested using SAS 9.4 (ODS ExcelXP tagset version 1.130) and Microsoft Excel 2010 software.

REQUIREMENTS

To use the techniques described in this paper, you must have the following software:

- Base SAS 9.1.3 Service Pack 4 or later on any supported operating system (z/OS, UNIX, etc.) and hardware.
- Microsoft Excel 2002 (also referred to as Microsoft Excel XP) or later.

LIMITATIONS

Because the ExcelXP ODS tagset creates files that conform to the Microsoft XML Spreadsheet Specification, you can create multi-sheet Excel workbooks that contain the output from almost any SAS procedure. The exception is that the Microsoft XML Spreadsheet Specification does not support images, so the output from graphics procedures cannot be used (Microsoft Corporation 2001).

You can use ExcelXP tagset options with all procedure output, but ODS style overrides apply only to the PRINT, REPORT, and TABULATE procedures. Tagset options and style overrides are discussed in the sections "Understanding and Using the ExcelXP Tagset Options" and "Understanding and Using ODS Style Overrides", respectively.

You cannot use the techniques described in this paper to update existing workbooks. ODS creates the entire document on each execution, and cannot alter existing workbooks.

SAMPLE DATA

Table 1 presents the column properties for the VaccineAE SAS table that is used to create the Excel workbook shown in Figure 1.

Column Name	Description	Typical Values
State2	2-digit state abbreviation	AK, DC, NC, WY
N2004 N2005 . . . N2013	Number of adverse events reports for 2004 to 2013	36, 817, 1144, 2906
Pct2004 Pct2005 . . . Pct2013	Percentage of total adverse events reports for 2004-2013	0.407602676, 1.3170141474, 10.193765796

Table 1. Column Properties and Representative Data Values for the VaccineAE SAS Table

The data was extracted from the Vaccine Adverse Event Reporting System (VAERS), available at http://vaers.hhs.gov. It is intended only for illustrative purposes.

OUTPUT DELIVERY SYSTEM (ODS) BASICS

ODS is the part of Base SAS software that enables you to generate different types of output from your procedure code. An ODS *destination* controls the type of output that is generated (HTML, RTF, PDF, etc.). An ODS *style* controls the appearance of the output. In this paper, we use a type of ODS destination, called a *tagset*, that creates XML output that can be opened with Excel. This tagset, named ExcelXP, creates an Excel workbook that has multiple worksheets.

The Excel workbook in Figure 1 was created using the ExcelXP ODS tagset and the PRINTER ODS style supplied by SAS. The ExcelXP tagset creates an XML file that, when opened by Excel, is rendered as a multi-sheet workbook. All formatting and layout are performed by SAS; there is no need to "hand-edit" the Excel workbook. You simply use Excel to open the file created by ODS.

Here are the general ODS statements to generate XML output that is compatible with Excel 2002 and later:

```
❶ ods _all_ close;

❷ ods tagsets.ExcelXP file='file-name.xml' style=style-name ... ;
    *  Your SAS procedure code here;
❸ ods tagsets.ExcelXP close;
```

The first ODS statement (❶) closes all destinations that are open because we want to generate only XML output for use with Excel.

The second ODS statement (❷) uses the ExcelXP tagset to generate the XML output and then store the output in a file. You should use the XML extension instead of XLSX or XLSX because Excel 2007 and later display a warning if the XML extension is not used (Microsoft Corporation 2015). The STYLE option controls the appearance of the output, such as the font and color scheme. To see a list of ODS styles that are available for use at your site, submit the following SAS code:

```
ods _all_ close;
ods listing;
proc template; list styles; run; quit;
```

To find the SAS code that generates sample output for the ODS styles available on your system, click the **Full Code** tab in SAS Sample 36900 (SAS Institute Inc. 2009).

The third ODS statement (❸) closes the ExcelXP destination and releases the XML file so that it can be opened with Excel.

Note: If you place the files where users can access them over a network, you should set file permissions to prevent accidental alteration.

OPENING THE OUTPUT WITH EXCEL

Follow these steps to open an ODS-generated XML file:

1. In Excel 2002, 2003, or 2010, select **`File`** ➡ **`Open`**.
   In Excel 2007 select **`Office Button`** ➡ **`Open`**.
2. Navigate to the file or enter the path and filename in the **`File name`** field.
3. Click **`Open`** to import the XML file.

You can also navigate to the file using Microsoft Windows Explorer, and then double-click the file to open it with Excel.

Excel reads and converts the XML file to the Excel format. After the conversion, you can perform any Excel function on the data. To save a copy of the file in Excel binary (XLS) format using Excel 2002, 2003, or 2010, select **File ➡ Save As** and then, from the **Save as type** drop-down list, select **Microsoft Excel Workbook (*.xls)**. If you're using Excel 2007, click the Microsoft Office Button, and then select **Save As ➡ Excel 97-2003 Workbook**. If you're using Excel 2007 or 2010 and want to save the document in the Microsoft Office Open XML format, choose **Excel Workbook (*.xlsx)** from the **Save as type** drop-down list.

UPDATING THE EXCELXP TAGSET

The version of the ExcelXP tagset that is shipped with Base SAS is periodically updated. There is currently no notification system for tagset updates. To ensure that you have a recent version, compare the ExcelXP tagset version, displayed in the SAS log whenever the tagset is used, to the version available on the ODS website (SAS Institute Inc. 2015).

Submit this code to display the tagset version number in the SAS log:

```
filename temp temp;

ods tagsets.ExcelXP file=temp;
ods tagsets.ExcelXP close;

filename temp clear;
```

All the code in this paper uses an up-to-date version of the ODS ExcelXP tagset (version 1.130). If you're using a tagset that's more than 2 or 3 versions old, consider upgrading by following the steps in SAS Usage Note 32394 (SAS Institute Inc. 2008b). Otherwise, continue to the next section.

USING ODS TO CREATE THE MULTI-SHEET EXCEL WORKBOOK

Here is a listing of the *basic* SAS code used to create the Excel workbook:

```
  ods _all_ close;

  options topmargin  = 0.5 in  bottommargin = 0.5 in
          leftmargin = 0.5 in  rightmargin  = 0.5 in;

❶ ods tagsets.ExcelXP file='VaccineAE.xml' style=Printer;

  title j=c 'Vaccine Adverse Event Reporting System (VAERS) Report for the
             United States';
  footnote j=l 'Source: http://vaers.hhs.gov';

  * First worksheet;

❷ proc report data=sample.VaccineAE nowd;

  column State2 ('2004' N2004 Pct2004)
                ('2005' N2005 Pct2005)
                ('2006' N2006 Pct2006)
                ('2007' N2007 Pct2007)
                ('2008' N2008 Pct2008);

  define State2  / display  'State';
  define N2004   / analysis 'N';
❸ define Pct2004 / analysis '(%)';
  define N2005   / analysis 'N';
  define Pct2005 / analysis '(%)';
  ... ;
  define N2008   / analysis 'N';
  define Pct2008 / analysis '(%)';

❹ rbreak after / summarize;

  run; quit;

  * Second worksheet;

  proc report data=sample.VaccineAE nowd;

  column State2 ('2009' N2009 Pct2009)
                ('2010' N2010 Pct2010)
                ('2011' N2011 Pct2011)
                ('2012' N2012 Pct2012)
                ('2013' N2013 Pct2013);

  define State2  / display  'State';
  define N2009   / analysis 'N';
  define Pct2009 / analysis '(%)';
  define N2010   / analysis 'N';
  define Pct2010 / analysis '(%)';
  ... ;
  define N2013   / analysis 'N';
  define Pct2013 / analysis '(%)';

  rbreak after / summarize;

  run; quit;

❺ ods tagsets.ExcelXP close;
```

Stepping through the code, all open ODS destinations are closed, and then margins are specified to control the appearance of the printed output.

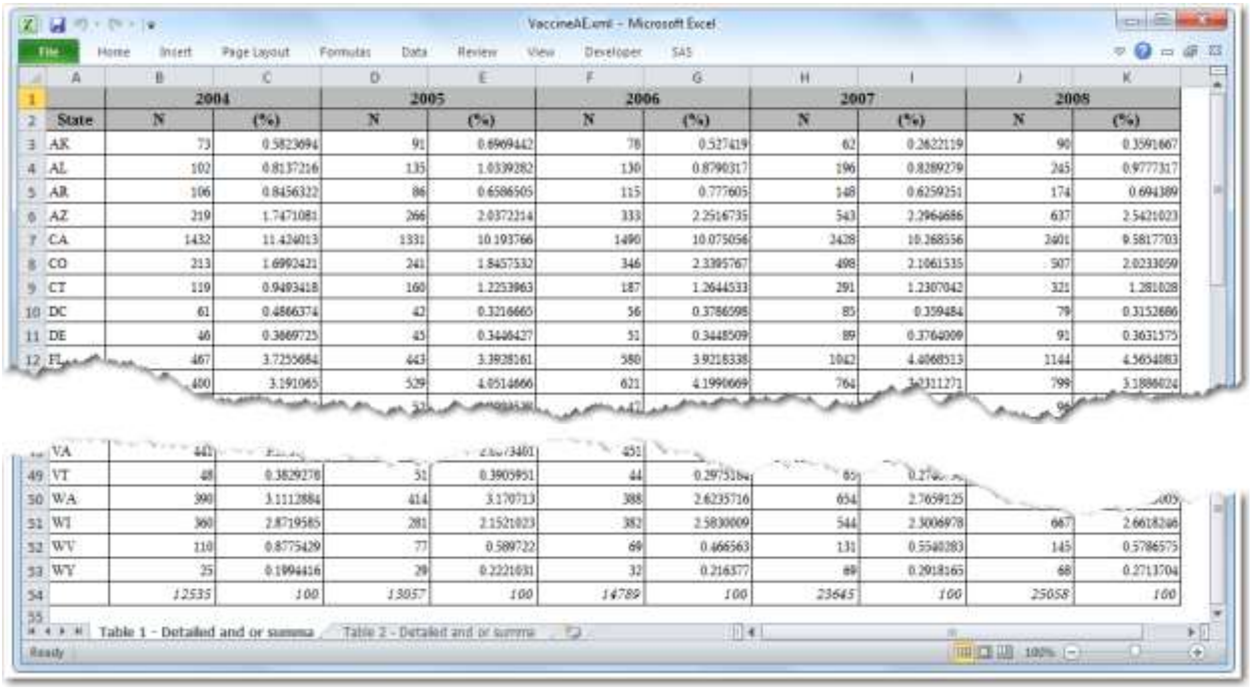
As you can see in the ODS statement (❶), the ExcelXP tagset generates the output and the PRINTER style controls the appearance of the output. By default, the ExcelXP tagset creates a new worksheet when a SAS procedure creates new tabular output. Each instance of the REPORT procedure (❷) creates one table showing half of the yearly data, and each table is created in a separate worksheet.

The RBREAK statement (❹) creates a summary line listing the sum of all the analysis variables. The last ODS statement (❺) closes the ExcelXP destination and releases the XML file so that it can be opened with Excel.

Figure 2 displays the results of executing the basic SAS code, and then opening the resulting VaccineAE.xml file with Excel. Notice that Figure 2 does not match Figure 1. The following problems are exhibited in Figure 2:

1. Unattractive, default worksheet names are used.
2. Title and footnote text are missing.
3. Printing results in more than one page per worksheet.
4. Incorrect display formats are used for the values in the numeric columns and summary row cells.
5. The background color of the summary row is not gray.

In the following sections we change the basic SAS code to correct these problems. The complete SAS code used to create the workbook shown in Figure 1 is listed in the section "The Final SAS Code".


VaccineAE.xml - Microsoft Excel
File
Home
Insert
Page Layout
Formulas
Data
Review
View
Developer
SAS
Table 1 - Detailed and or summa
Table 2 - Detailed and or summa
Ready
100%

	2004		2005		2006		2007		2008	
State	N	(%)	N	(%)	N	(%)	N	(%)	N	(%)
AK	73	0.5823694	91	0.6969442	78	0.527419	62	0.2622119	90	0.3591667
AL	102	0.8137216	135	1.0339282	130	0.8790317	196	0.8289279	245	0.9777317
AR	106	0.8456322	86	0.6586505	115	0.777605	148	0.6259251	174	0.694389
AZ	219	1.7471081	266	2.0372214	333	2.2516735	543	2.2964686	637	2.5421023
CA	1432	11.424013	1331	10.193766	1490	10.075056	2428	10.268556	2401	9.5817703
CO	213	1.6992421	241	1.8457532	346	2.3395767	498	2.1061535	507	2.0233059
CT	119	0.9493418	160	1.2253963	187	1.2644533	291	1.2307042	321	1.281028
DC	61	0.4866374	42	0.3216665	56	0.3786598	85	0.359484	79	0.3152686
DE	46	0.3669725	45	0.3446427	51	0.3448509	89	0.3764009	91	0.3631575
FL	467	3.7255684	443	3.3928161	580	3.9218338	1042	4.4068513	1144	4.5654083
[illegible]	400	3.191065	529	4.0514666	621	4.1990669	764	3.2311271	799	3.1888024
[illegible]	[illegible]	[illegible]	53	[illegible]	47	[illegible]	[illegible]	[illegible]	96	[illegible]
VA	441	[illegible]	[illegible]	[illegible]	451	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
VT	48	0.3829278	51	0.3905951	44	0.2975184	65	[illegible]	[illegible]	[illegible]
WA	390	3.1112884	414	3.170713	388	2.6235716	654	2.7659125	[illegible]	[illegible]
WI	360	2.8719585	281	2.1521023	382	2.5830009	544	2.3006978	667	2.6618246
WV	110	0.8775429	77	0.589722	69	0.466563	131	0.5540283	145	0.5786575
WY	25	0.1994416	29	0.2221031	32	0.216377	69	0.2918165	68	0.2713704
	12535	*100*	*13057*	*100*	*14789*	*100*	*23645*	*100*	*25058*	*100*

Figure 2. Initial ODS ExcelXP Tagset-Generated Workbook

UNDERSTANDING AND USING THE EXCELXP TAGSET OPTIONS

The ExcelXP tagset supports many options that control both the appearance and functionality of the Excel workbook. Many of these tagset options are simply tied directly into existing Excel options or features. For example, the SHEET_NAME option is used to specify the worksheet name.

Tagset options are specified in an ODS statement using the OPTIONS keyword:

```
ods tagsets.ExcelXP options(option-name1='value1'
                            option-name2='value2' ...) ... ;
```

Note that the value that you specify for a tagset option remains in effect until the ExcelXP destination is closed or the option is set to another value. Because multiple ODS statements are allowed, it is good practice, in terms of functionality and code readability, to explicitly reset tagset options to their default values when you are finished using them.

For example:

```
ods tagsets.ExcelXP file='file-name.xml' style=style-name ... ;
  ods tagsets.ExcelXP options(option-name='some-value');
    *  Some SAS procedure code here;
  ods tagsets.ExcelXP options(option-name='default-value');
    *  Other SAS procedure code here;
ods tagsets.ExcelXP close;
```

When specifying multiple ODS statements as shown above, specify the FILE, STYLE, or any other keyword or option that is supported by ODS only in the initial ODS statement.

To see a listing of the supported options printed to the SAS log, submit the following SAS code:

```
filename temp temp;

ods tagsets.ExcelXP file=temp options(doc='help');
ods tagsets.ExcelXP close;

filename temp clear;
```

All of the tagset options and code work with any SAS procedure. However, ODS style overrides work only with the PRINT, REPORT, and TABULATE procedures.

SPECIFYING WORKSHEET NAMES

ODS generates a unique name for each worksheet, as required by Excel. Figure 2 shows the worksheet names that result from running the initial SAS code. There are, however, several tagset options that you can use to alter the names of the worksheets (DelGobbo 2013, 2014).

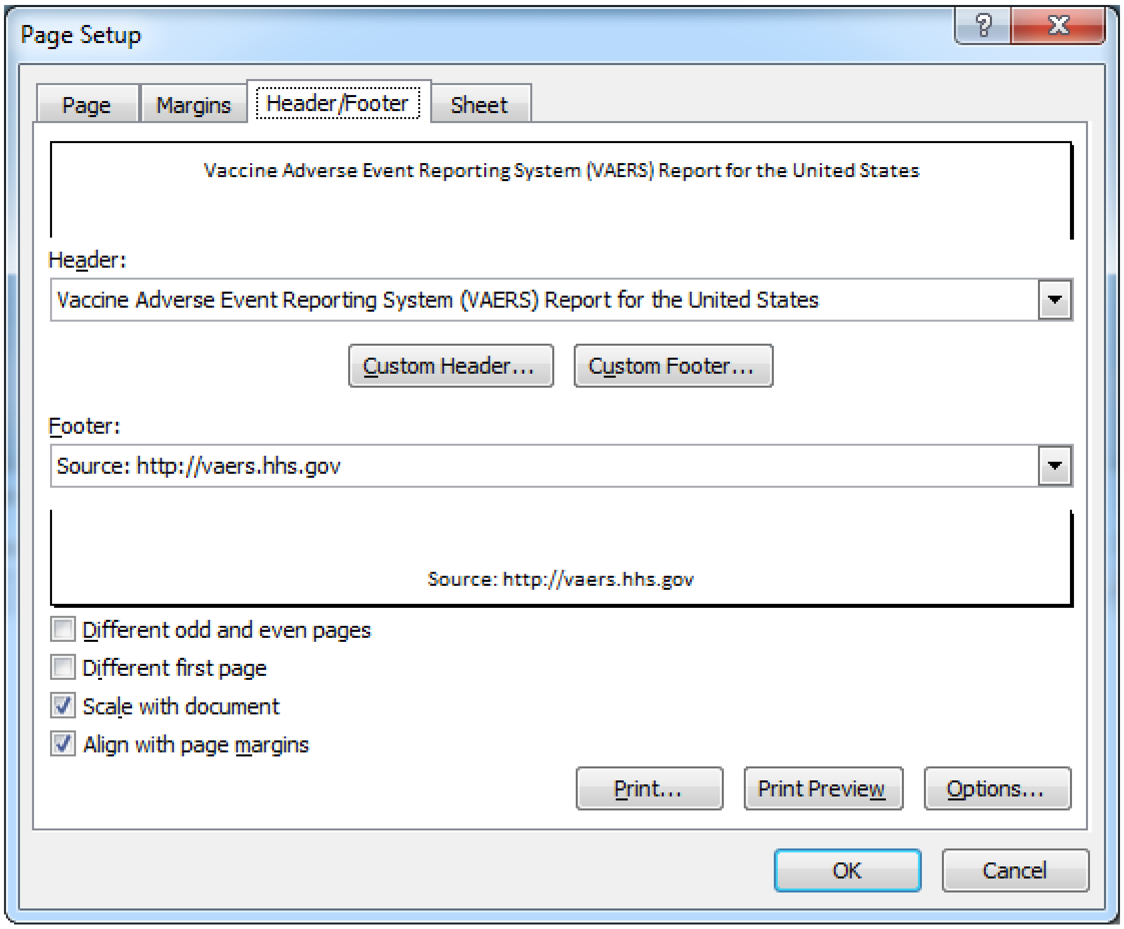
Use the SHEET_NAME option to specify a worksheet name. Recall that tagset options remain in effect until the ExcelXP destination is closed. We specify the option twice because we want a different name for each worksheet.

```
ods tagsets.ExcelXP file='VaccineAE.xml' style=Printer;

title ... ; footnote ... ;

* First worksheet;

ods tagsets.ExcelXP options(sheet_name='2004 - 2008');

proc report data=sample.VaccineAE nowd;
... ;
run; quit;

* Second worksheet;

ods tagsets.ExcelXP options(sheet_name='2009 - 2013');

proc report data=sample.VaccineAE nowd;
... ;
run; quit;

ods tagsets.ExcelXP close;
```

Figure 4 shows the updated worksheet names.

INCLUDING TITLE AND FOOTNOTE TEXT IN THE WORKSHEET BODY

By default, SAS titles and footnotes appear as Excel print headers and print footers, respectively, which are displayed when the Excel document is printed. You can confirm this by viewing the Excel **Header/Footer** tab in the Page Setup dialog box, shown in Figure 3.



Vaccine Adverse Event Reporting System (VAERS) Report for the United States

Header: Vaccine Adverse Event Reporting System (VAERS) Report for the United States

Footer: Source: http://vaers.hhs.gov

Source: http://vaers.hhs.gov

- [ ] Different odd and even pages
- [ ] Different first page
- [x] Scale with document
- [x] Align with page margins

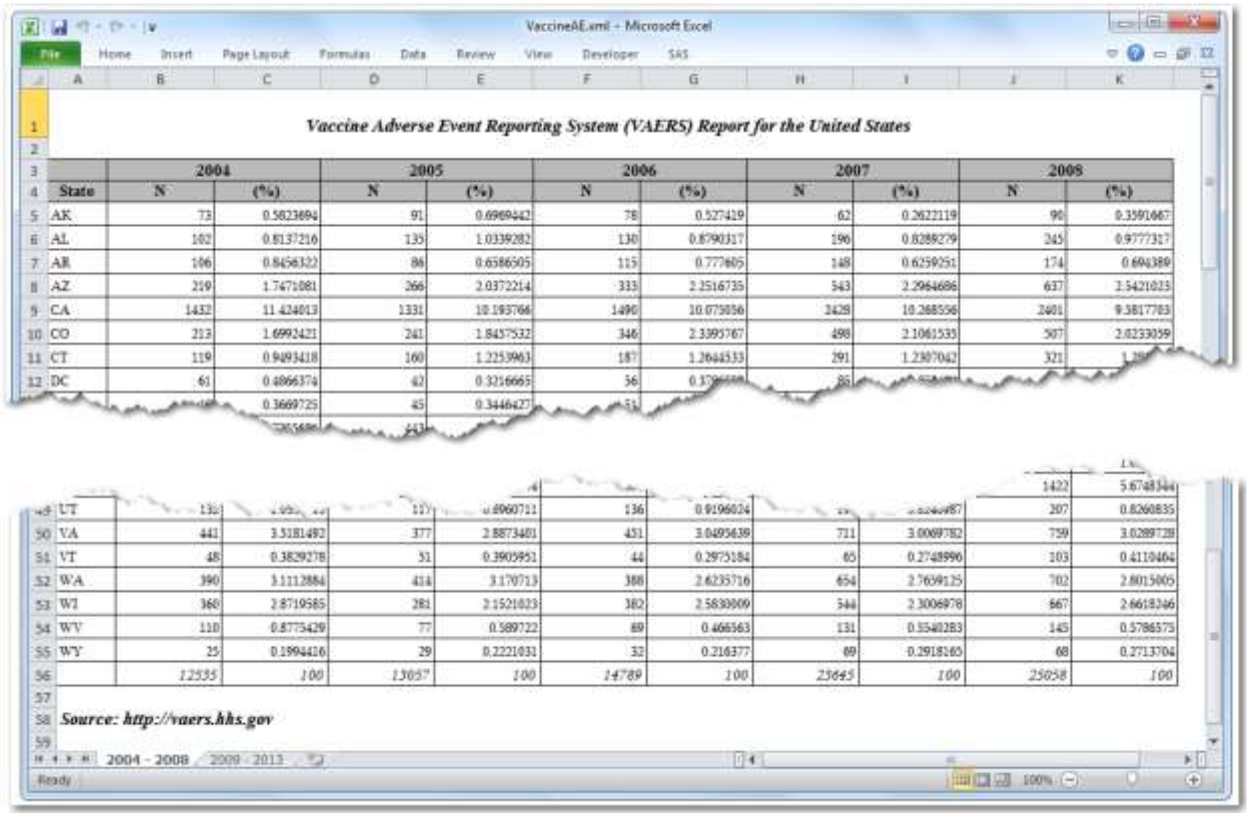
Figure 3. Excel Page Setup Dialog Box Showing Title and Footnote Text in Headers

To include title and footnote text on-screen, in the worksheet body, use the EMBEDDED_TITLES and EMBEDDED_FOOTNOTES options:

```
ods tagsets.ExcelXP file='VaccineAE.xml' style=Printer;

*  "Global" tagset options;

ods tagsets.ExcelXP options(embedded_titles='yes'
                            embedded_footnotes='yes');

title ... ; footnote ... ;

* First worksheet;

ods tagsets.ExcelXP options(sheet_name='2004 - 2008');

proc report data=sample.VaccineAE nowd;
... ;
run; quit;
```

```
* Second worksheet;

ods tagsets.ExcelXP options(sheet_name='2009 - 2013');

proc report data=sample.VaccineAE nowd;
... ;
run; quit;

ods tagsets.ExcelXP close;
```

Because these options are placed at the beginning of the code and are not changed later, they affect all worksheets. The title and footnote text are now included in the worksheet body (Figure 4).



Vaccine Adverse Event Reporting System (VAERS) Report for the United States

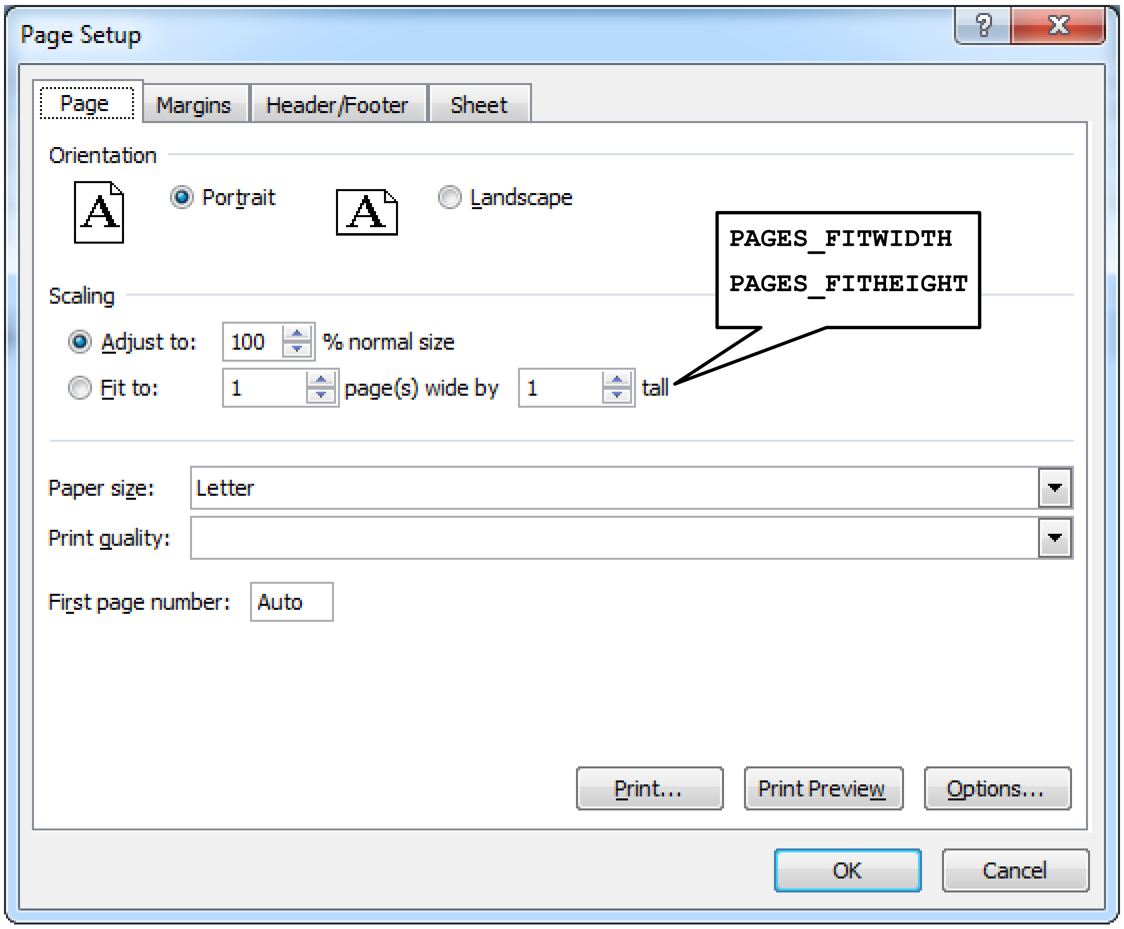
	2004		2005		2006		2007		2008	
State	N	(%)	N	(%)	N	(%)	N	(%)	N	(%)
AK	73	0.5823694	91	0.6969442	78	0.527419	62	0.2622119	90	0.3591667
AL	102	0.8137216	135	1.0339282	130	0.8790317	196	0.8289279	245	0.9777317
AR	106	0.8456322	86	0.6586505	115	0.777605	148	0.6259251	174	0.694389
AZ	219	1.7471081	266	2.0372214	333	2.2516735	543	2.2964686	637	2.5421023
CA	1432	11.424013	1331	10.193766	1490	10.075056	2428	10.268556	2401	9.5817783
CO	213	1.6992421	241	1.8457532	346	2.3395767	498	2.1061535	507	2.0233059
CT	119	0.9493418	160	1.2253963	187	1.2644533	291	1.2307042	321	[illegible]
DC	61	0.4866374	42	0.3216665	56	[illegible]	86	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	0.3669725	45	0.3446427	51	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1422	5.6748344
UT	132	[illegible]	117	0.8960711	136	0.9196024	[illegible]	[illegible]	207	0.8260835
VA	441	3.5181492	377	2.8873401	451	3.0495639	711	3.0069782	759	3.0289728
VT	48	0.3829278	51	0.3905951	44	0.2975184	65	0.2748996	103	0.4110464
WA	390	3.1112884	414	3.170713	388	2.6235716	654	2.7659125	702	2.8015005
WI	360	2.8719585	281	2.1521023	382	2.5830009	544	2.3006978	667	2.6618246
WV	110	0.8775429	77	0.589722	69	0.466563	131	0.5540283	145	0.5786575
WY	25	0.1994416	29	0.2221031	32	0.216377	69	0.2918165	68	0.2713704
	12535	*100*	*13057*	*100*	*14789*	*100*	*23645*	*100*	*25058*	*100*

Source: http://vaers.hhs.gov

Figure 4. ODS ExcelXP Tagset-Generated Workbook with Title, Footnote, and Worksheet Names

CONSTRAINING THE PRINTED OUTPUT TO A SINGLE PAGE PER WORKSHEET

One of the problems with the initial output is that it exceeds more than one page when a worksheet is printed. Figure 5 and Figure 6 show the Excel settings and the corresponding ExcelXP tagset options that constrain the printed output to one page per worksheet.



Page Setup

Page | Margins | Header/Footer | Sheet

Orientation

[✓ Portrait] Landscape

Scaling

[✓ Adjust to:] 100 % normal size

Fit to: 1 page(s) wide by 1 tall

Paper size: Letter

Print quality:

First page number: Auto

Figure 5. Excel Page Setup Dialog Box Showing Page Options

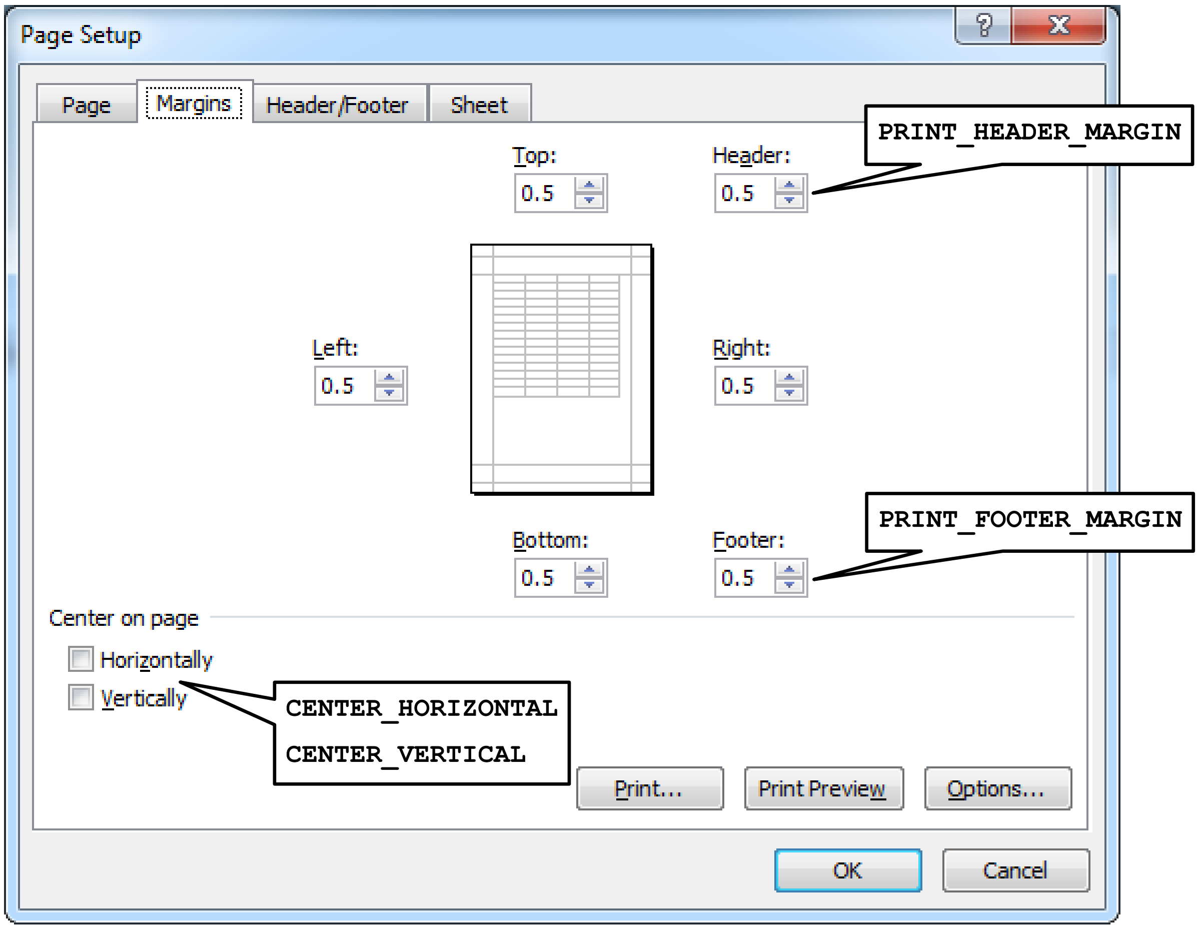


Figure 6. Excel Page Setup Dialog Box Showing Margin Settings

The remaining margin specifications in Figure 6 are controlled by the OPTIONS statement shown in the "Using ODS to Create the Multi-Sheet Excel Workbook" section.

Here is the code to center the printed output horizontally and vertically on a single page:

```
ods tagsets.ExcelXP file='VaccineAE.xml' style=Printer;

*  "Global" tagset options;

ods tagsets.ExcelXP options(embedded_titles='yes'
                            embedded_footnotes='yes'
                            pages_fitwidth='1'
                            pages_fitheight='1'
                            print_header_margin='0'
                            print_footer_margin='0'
                            center_horizontal='yes'
                            center_vertical='yes');

title ... ; footnote ... ;

* First worksheet;

ods tagsets.ExcelXP options(sheet_name='2004 - 2008');
```

```
proc report data=sample.VaccineAE nowd;
... ;
run; quit;

* Second worksheet;

ods tagsets.ExcelXP options(sheet_name='2009 - 2013');

proc report data=sample.VaccineAE nowd;
... ;
run; quit;

ods tagsets.ExcelXP close;
```

The printed output from each worksheet now fits on a single page in portrait orientation.

UNDERSTANDING AND USING ODS STYLE OVERRIDES

You can alter the attributes or style elements used by specific parts of your SAS output by using style overrides. These specific parts of your SAS output are called *locations*. Figure 7 shows the locations that are pertinent to the REPORT procedure output (SAS Institute Inc. 2008a). The COLUMN location controls the appearance of data cells and the SUMMARY location controls the summary line cells.

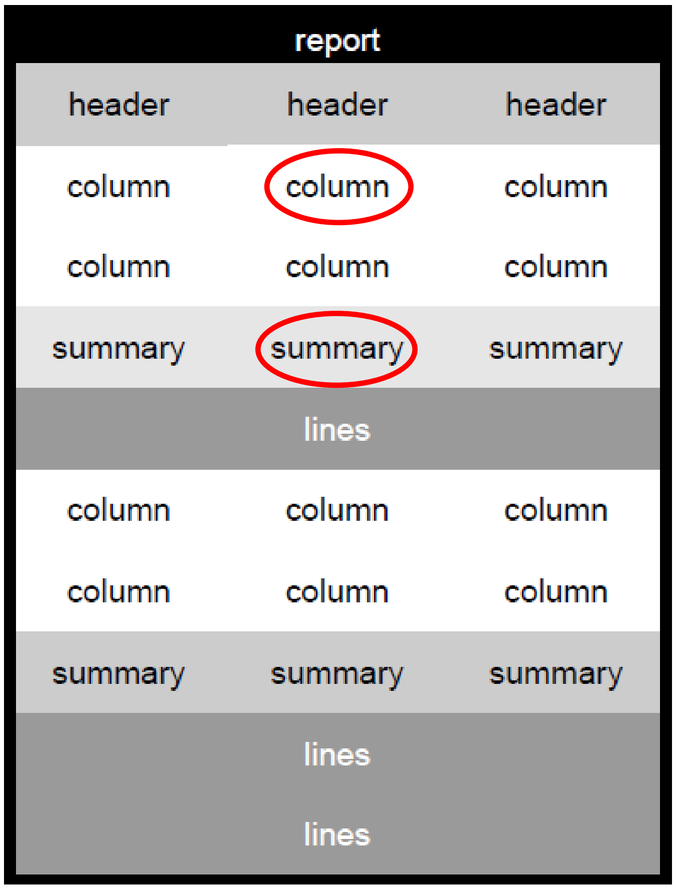


Figure 7. Style Locations for the REPORT Procedure

Style overrides are supported by the PRINT, REPORT, and TABULATE procedures, and can be specified several ways. Here are the two most common formats:

```
❶ style(location)=[style-attribute-name1=value1
                    style-attribute-name2=value2 ...]

❷ style(location)=style-element-name
```

The first format (❶) uses individual style attributes defined inline. For example, this PROC REPORT code alters three attributes of the COLUMN location for the MYVAR variable:

```
define myvar / style(column)=[background=yellow font_size=10pt just=left];
```

Although this is the most commonly used format, it has some disadvantages. To use the same style override for different variables, you must apply it in multiple places, making your SAS code harder to read and maintain. And, if you want to use the style overrides in other SAS programs, you must copy the list of attribute name/value pairs to the new code. Because of these drawbacks, inline style overrides should be used sparingly.

The second format (❷) overcomes these problems by referencing a style element. Using this format involves creating a new style element, setting the style attributes within the element, and then using the *style element* name in your style override. This results in code that is easier to read, maintain, and reuse. Earlier papers by this author provide a detailed discussion of this topic (DelGobbo 2008, 2009, 2010, 2011).

Refer to the ODS documentation for a full listing of style attributes (SAS Institute Inc. 2014a).

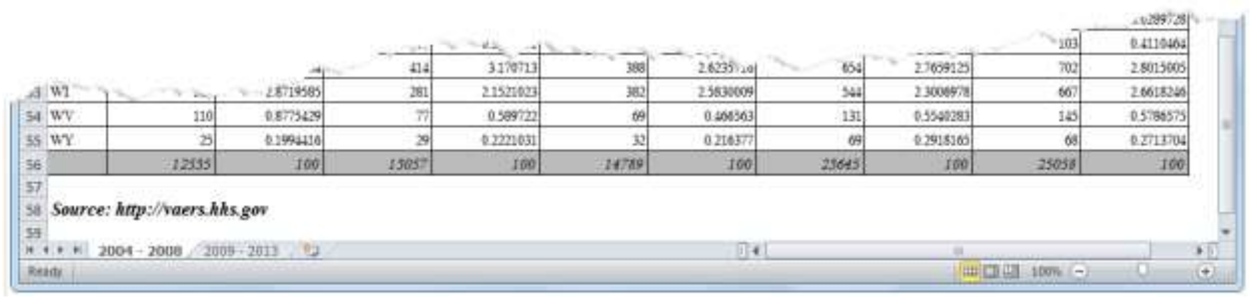
CHANGING THE BACKGROUND COLOR OF THE SUMMARY LINE

We want a gray background color for the summary line to help draw attention to the column totals. To accomplish this, change the value of the BACKGROUND attribute for the SUMMARY location using a style override:

```
ods tagsets.ExcelXP file='VaccineAE.xml' style=Printer;

*  "Global" tagset options;

ods tagsets.ExcelXP options( ... );

title ... ; footnote ... ;

* First worksheet;

ods tagsets.ExcelXP options(sheet_name='2004 - 2008');

proc report data=sample.VaccineAE nowd;

column ... ;
define ... ;

rbreak after / summarize style(summary)=[background=#bbbbbb];

run; quit;

* Second worksheet;

ods tagsets.ExcelXP options(sheet_name='2009 - 2013');
```

```
proc report data=sample.VaccineAE nowd;

column ... ;
define ... ;

rbreak after / summarize style(summary)=[background=#bbbbbb];

run; quit;

ods tagsets.ExcelXP close;
```

Note that no changes were made to the ODS ExcelXP tagset options that were previously specified.   The results are shown in Figure 8.



											0289728
										103	0.4110464
				414	3.170713	388	2.6235	654	2.7659125	702	2.8015005
53	WI		2.8719585	281	2.1521023	382	2.5830009	544	2.3008978	667	2.6618246
54	WV	110	0.8775429	77	0.589722	69	0.466563	131	0.5540283	145	0.5786575
55	WY	25	0.1994416	29	0.2221031	32	0.216377	69	0.2918165	68	0.2713704
56		*12535*	*100*	*15057*	*100*	*14789*	*100*	*23645*	*100*	*25058*	*100*
57											
58	***Source: http://vaers.hhs.gov***										
59											

Figure 8. ODS ExcelXP Tagset-Generated Workbook with Gray Background in Summary Line

APPLYING AN EXCEL FORMAT TO THE ANALYSIS COLUMNS

Our Excel workbook now closely resembles Figure 1, except that the display formats for the analysis columns and summary line do not match. Although you might (naturally) want to use SAS formats to control the Excel display values, it's better to use Excel formats because they're always honored.

Figure 9 shows the general structure of Excel number formats (Microsoft Corporation 2014). The pound sign (#) in an Excel format represents a numeric digit, excluding insignificant zeros. A zero (0) displays a numeric digit, including insignificant zeros. Use zeros in Excel formats when you want to retain leading or trailing zeros.

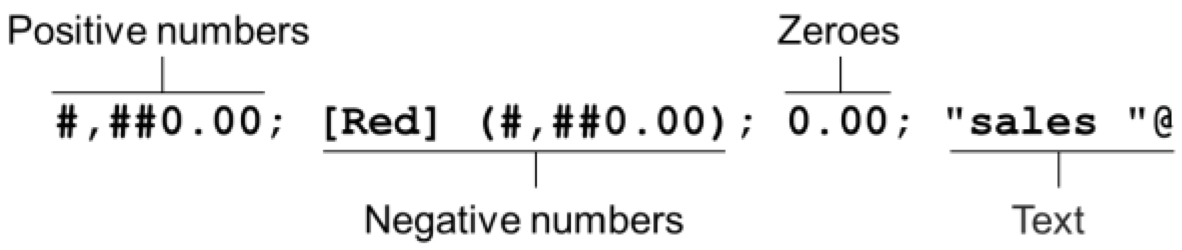


Figure 9. Structure of Excel Number Formats

Table 2 shows the results of applying the Excel format shown in Figure 9.

Raw Value	Formatted Value	Comment
.5	0.50	Leading and trailing zeros
5	5.00	Leading and trailing zeros
123	123.00	Trailing zeros
1234	1,234.00	Trailing zeros and thousands separator
-1234	(1,234.00)	Leading and trailing zeros, thousands separator, and red ()
0	0.00	Special zero handling
data	sales data	Special text handling

Table 2. Results of Applying Excel Format Shown in Figure 9

Because all of the values that we need to format are positive numbers, we specify only the first section of the Excel format. The zeros insure that the formatted values contain a value between 0 and 9 in that position.

```
* First worksheet;

ods tagsets.ExcelXP options(sheet_name='2004 - 2008');

proc report data=sample.VaccineAE nowd;

column ... ;

define State2  / display  'State';

define N2004   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2004 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];
... ;
define N2008   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2008 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];

rbreak after / summarize style(summary)=[background=#bbbbbb];

run; quit;

* Second worksheet;

ods tagsets.ExcelXP options(sheet_name='2009 - 2013');

proc report data=sample.VaccineAE nowd;

column ... ;

define State2  / display  'State';

define N2009   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2009 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];
... ;
define N2013   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2013 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];

rbreak after / summarize style(summary)=[background=#bbbbbb];

run; quit;
```

With all of the code modifications in place, the resulting workbook matches the output shown in Figure 1.

THE FINAL SAS CODE

The final SAS code to create the output of Figure 1 follows:

```
ods _all_ close;

options topmargin  = 0.5 in  bottommargin = 0.5 in
        leftmargin = 0.5 in  rightmargin  = 0.5 in;

ods tagsets.ExcelXP file='VaccineAE-Final.xml' style=Printer;

*  "Global" tagset options;

ods tagsets.ExcelXP options(embedded_titles='yes'
                            embedded_footnotes='yes'
                            pages_fitwidth='1'
                            pages_fitheight='1'
                            print_header_margin='0'
                            print_footer_margin='0'
                            center_horizontal='yes'
                            center_vertical='yes');

title j=c 'Vaccine Adverse Event Reporting System (VAERS) Report for the
United States';
footnote j=l 'Source: http://vaers.hhs.gov';

* First worksheet;

ods tagsets.ExcelXP options(sheet_name='2004 - 2008');

proc report data=sample.VaccineAE nowd;

column State2 ('2004' N2004 Pct2004)
              ('2005' N2005 Pct2005)
              ('2006' N2006 Pct2006)
              ('2007' N2007 Pct2007)
              ('2008' N2008 Pct2008);

define State2  / display  'State';
define N2004   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2004 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];
define N2005   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2005 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];
define N2006   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2006 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];
define N2007   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2007 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];
define N2008   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2008 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];

rbreak after / summarize style(summary)=[background=#bbbbbb];

run; quit;
```

```
* Second worksheet;

ods tagsets.ExcelXP options(sheet_name='2009 - 2013');

proc report data=sample.VaccineAE nowd;

column State2 ('2009' N2009 Pct2009)
              ('2010' N2010 Pct2010)
              ('2011' N2011 Pct2011)
              ('2012' N2012 Pct2012)
              ('2013' N2013 Pct2013);

define State2  / display  'State';
define N2009   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2009 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];
define N2010   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2010 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];
define N2011   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2011 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];
define N2012   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2012 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];
define N2013   / analysis 'N'   style(column)=[tagattr='format:#,##0'];
define Pct2013 / analysis '(%)' style(column)=[tagattr='format:(0.0)'];

rbreak after / summarize style(summary)=[background=#bbbbbb];

run; quit;

ods tagsets.ExcelXP close;
```

SAS SERVER TECHNOLOGY

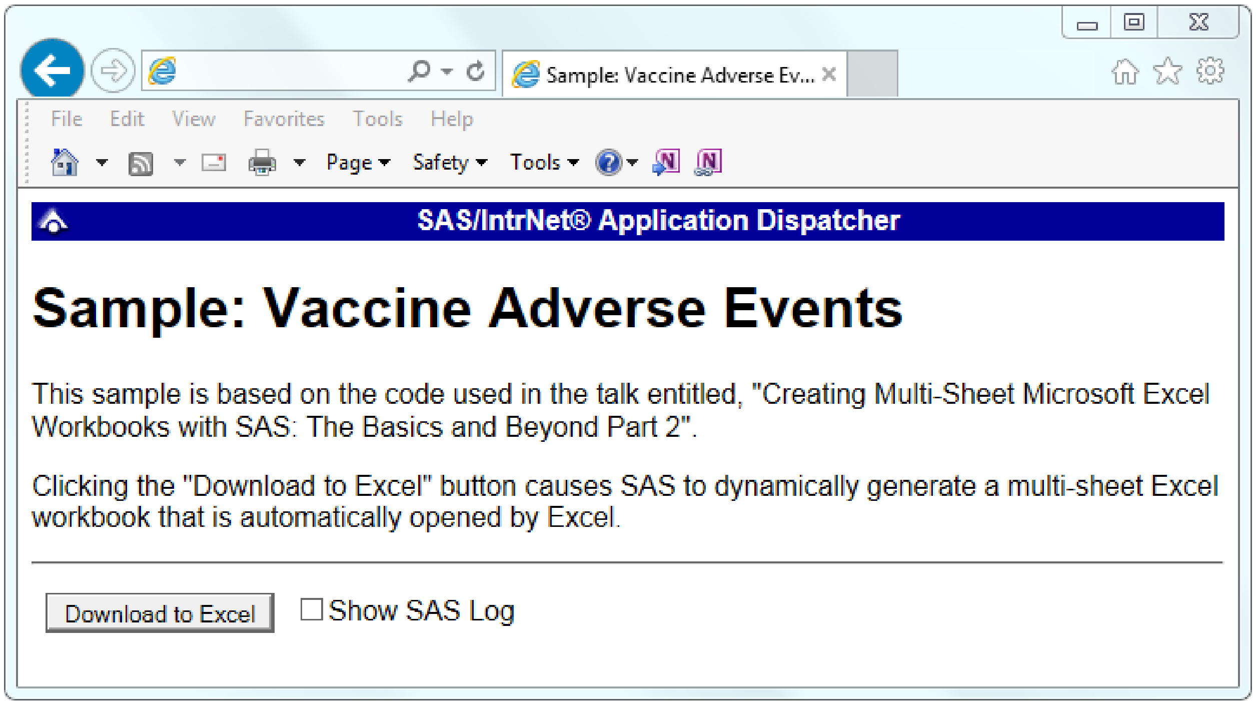
You can deliver dynamically generated SAS output into Excel using the Application Dispatcher or the SAS® Stored Process Server. The Application Dispatcher is part of SAS/IntrNet® software. The SAS Stored Process Server is available starting with SAS®9 as part of SAS® Integration Technologies, and is included with server offerings that leverage the SAS Business Analytics infrastructure (for example, SAS® BI Server and SAS® Enterprise BI Server).

These products enable you to execute SAS programs from a Web browser or any other client that can open an HTTP connection to the Application Dispatcher or the SAS Stored Process Server. Both of these products can run on any platform where SAS is licensed. SAS software does not need to be installed on the client machine.

The SAS programs that you execute from the browser can contain any combination of DATA step, procedure, macro, or SCL code. Thus, all of the code that has been shown up to this point can be executed by both the Application Dispatcher and the SAS Stored Process Server.

Program execution is typically initiated by accessing a URL that points to the SAS server program. Parameters are passed to the program as name/value pairs in the URL. The SAS server takes these name/value pairs and constructs SAS macro variables that are available to the SAS program.

Figure 10 shows a Web page that can deliver SAS output directly to Excel, using a Web browser as the client.


Sample: Vaccine Adverse Ev...
File
Edit
View
Favorites
Tools
Help
Page
Safety
Tools
SAS/IntrNet® Application Dispatcher

Sample: Vaccine Adverse Events

This sample is based on the code used in the talk entitled, "Creating Multi-Sheet Microsoft Excel Workbooks with SAS: The Basics and Beyond Part 2".

Clicking the "Download to Excel" button causes SAS to dynamically generate a multi-sheet Excel workbook that is automatically opened by Excel.

Figure 10. Web Page to Drive a SAS/IntrNet Application

Clicking **Download to Excel** executes a slightly modified version of the SAS code that we have been working on. The modifications are as follows:

```
%let RV=%sysfunc(appsrv_header(Content-type,application/vnd.ms-excel));
%let RV=%sysfunc(appsrv_header(Content-disposition,attachment; filename=
"VaccineAE.xml"));  * Ignore line wrapping;

ods _all_ close;

options topmargin=0.5 in ... ;

ods tagsets.ExcelXP file=_webout style=Printer;

*  Remainder of the "final" SAS code;

ods tagsets.ExcelXP close;
```

The first APPSRV_HEADER function sets a MIME header that causes the SAS output to be opened by Excel, instead of being rendered by the Web browser. This statement is required.

The second APPSRV_HEADER function sets a MIME header that causes the filename to be displayed in the File Download dialog box. As you can see in Figure 11, the filename appears as **VaccineAE.xml**. This header might cause problems with some versions of Excel, so be sure to test your applications before deploying them in a production environment. This statement is optional.

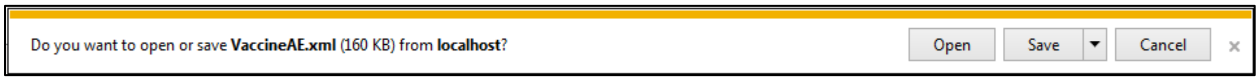

Do you want to open or save **VaccineAE.xml** (160 KB) from **localhost**?

Figure 11. File Download Dialog Box

The reserved fileref _WEBOUT is automatically defined by the SAS server and is always used to direct output from the SAS server to the client. Modify your existing ODS statement to direct the output to this fileref instead of to an external disk file.

When you click the **Download to Excel** button on the Web page you are presented with the File Download dialog box (Figure 11). You can then click **Open** to immediately open your SAS output using Excel, or you can click **Save**.

This is only one example of how you can dynamically deliver SAS output to Excel. For more detailed information and other examples, see the SAS/IntrNet Application Dispatcher and SAS Stored Process documentation (SAS Institute Inc. 2013, 2014b), as well as this author's earlier papers (DelGobbo 2002, 2003, 2004).

CONCLUSION

The SAS ExcelXP ODS tagset provides an easy way to export your SAS data to Excel workbooks that contain multiple worksheets. By using ODS styles, style overrides, and a tagset that complies with the Microsoft XML Spreadsheet Specification, you can customize the output to achieve your design goals.

APPENDIX

VISUAL BASIC CODE TO CONVERT XML TO NATIVE EXCEL FORMATS

The author is developing a Visual Basic script that converts ExcelXP-generated files to native Excel XLS or XLSX formats. Contact the author if you would like a copy of this experimental code.

FUTURE DIRECTION

SAS Institute continues to work toward better Microsoft Office integration, and future releases of SAS software will provide even more robust means of using SAS content with Microsoft Office applications. Upcoming maintenance releases of SAS 9.4 will include a new ODS destination to create native XLSX files (including graphics) and a procedure to create native Excel charts.

REFERENCES

DelGobbo, Vincent. 2002. "Techniques for SAS® Enabling Microsoft® Office in a Cross-Platform Environment". *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. Available at http://www2.sas.com/proceedings/sugi27/p174-27.pdf.

DelGobbo, Vincent. 2003. "A Beginner's Guide to Incorporating SAS® Output in Microsoft® Office Applications". *Proceedings of the Twenty-Eighth Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. Available at http://www2.sas.com/proceedings/sugi28/052-28.pdf.

DelGobbo, Vincent. 2004. "From SAS® to Excel via XML".
Available at http://support.sas.com/rnd/papers/sugi29/ExcelXML.pdf.

DelGobbo, Vincent. 2008. "Tips and Tricks for Creating Multi-Sheet Microsoft Excel Workbooks the Easy Way with SAS®". *Proceedings of the SAS Global Forum 2008 Conference*. Cary, NC: SAS Institute Inc. Available at http://www2.sas.com/proceedings/forum2008/192-2008.pdf.

DelGobbo, Vincent. 2009. "More Tips and Tricks for Creating Multi-Sheet Microsoft Excel Workbooks the Easy Way with SAS®". *Proceedings of the SAS Global Forum 2009 Conference*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings09/152-2009.pdf.

DelGobbo, Vincent. 2010. "Traffic Lighting Your Multi-Sheet Microsoft Excel Workbooks the Easy Way with SAS®". *Proceedings of the SAS Global Forum 2010 Conference*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings10/153-2010.pdf.

DelGobbo, Vincent. 2013. "Some Techniques for Integrating SAS® Output with Microsoft Excel Using Base SAS®". *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings13/143-2013.pdf.

DelGobbo, Vincent. 2014. "Creating Multi-Sheet Microsoft Excel Workbooks with SAS®: The Basics and Beyond Part 1". *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/resources/papers/proceedings14/SAS050-2014.pdf.

Microsoft Corporation. 2001. "XML Spreadsheet Reference".
Available at http://msdn.microsoft.com/en-us/library/aa140066(office.10).aspx.

Microsoft Corporation. 2014. "Number format codes". Available at https://support.office.com/en-us/article/Number-format-codes-5026bbd6-04bc-48cd-bf33-80f18b4eae68.

Microsoft Corporation. 2015. "When you open a file in Excel 2007, you receive a warning that the file format differs from the format that the file name extension specifies". Available at http://support.microsoft.com/kb/948615.

SAS Institute Inc. 2008a. "SAS® 9 Reporting Procedure Styles Tip Sheet".
Available at http://support.sas.com/rnd/base/ods/scratch/reporting-styles-tips.pdf.

SAS Institute Inc. 2008b. "Usage Note 32394: Installing and Storing Updated Tagsets for ODS MARKUP". Available at http://support.sas.com/kb/32/394.html.

SAS Institute Inc. 2009. "Sample 36900: Instructions for viewing all of the style templates that are shipped with the SAS® System". Available at http://support.sas.com/kb/36/900.html.

SAS Institute Inc. 2013. *SAS/IntrNet® 9.4: Application Dispatcher.* Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/cdl/en/dispatch/64895/HTML/default/viewer.htm#p06h82ux8glu1pn16k9dxw8tjpyz.htm.

SAS Institute Inc. 2014a. *SAS® 9.4 Output Delivery System: User's Guide, Third Edition.* Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/cdl/en/odsug/67325/HTML/default/viewer.htm#p0xi2cygmfk0wkn1ei625zq5r488.htm.

SAS Institute Inc. 2014b. *SAS® 9.4 Stored Processes: Developer's Guide*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/cdl/en/stpug/67499/HTML/default/viewer.htm#n180km1hadyuton13gx4bzlqhpwr.htm.

SAS Institute Inc. 2015. "ODS MARKUP Resources".
Available at http://support.sas.com/rnd/base/ods/odsmarkup.

ACKNOWLEDGMENTS

The author would like to thank Chris Barrett of SAS Institute for his valuable contributions to this paper.

RECOMMENDED READING

SAS Institute Inc. 2015. "Vince DelGobbo's ExcelXP Tagset Paper Index". Available at http://support.sas.com/community/events/sastalks/presentations/ExcelXPPaperIndex.pdf.

CONTACT INFORMATION

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

Vincent DelGobbo
SAS Institute Inc.
100 SAS Campus Drive
Cary, NC 27513

sasvcd@SAS.com

http://www.sas.com/reg/gen/corp/867226?page=Resources

If your registered in-house or local SAS users group would like to request this presentation as your annual SAS presentation (as a seminar, talk, or workshop) at an upcoming meeting, please submit an online User Group Request Form (from support.sas.com/sasusersupport/usergroups/support) at least eight weeks in advance.