Paper HW-06

SAS® to Publishable Excel... Seamlessly – Using ODS, XML, and Other Tricks

Harry Droogendyk, Stratia Consulting Inc.
Marje Fecht, Prowerk Consulting Ltd

ABSTRACT

Do you have legacy "listing" reports that you want to publish easily in Excel? Do you currently output to Excel but dislike having to massage the output to get things just right?

This hands-on workshop will introduce you to XML, and show you how to leverage the strengths of ODS to generate custom-formatted output in Excel. Topics include:

- creating Excel files using DATA _NULL_
- using SAS's ODS ExcelXP tagsets to *easily* customize Excel document properties
- using SAS to create CSV and XML data to feed Excel pivot tables

This workshop will prepare you to use ODS for creating Excel reports that are ready to publish with minimal (or no) post-processing.

INTRODUCTION

There are an abundance of tools and approaches to moving data from SAS into Excel. Each approach has an *intended purpose* along with a long list of advantages and disadvantages. Since the authors understand and recognize (from personal experience) that time and resources don't always permit a total rewrite of ***production code that is working just fine***, the focus of this paper is to help you take **existing legacy code** and make the additions and alterations necessary to publish your results in Excel. Further, our goal is to **avoid** much of the **post-processing** sometimes required to make the Excel output acceptable to the information consumer.

Most of the Excel content created by the examples in this paper is accessible in Excel 2000 or later. However, where features are only available in Excel 2002 or Excel 2003, they will be noted in the paper.

GOOD OLE DATA _NULL_

I bet you have **DATA _NULL_** code lurking in some of your production processes, and it has probably been running for years to produce reports that are widely used throughout your organization. I bet you have also occasionally tweaked the code to make the output just a little more functional and a little bit prettier. Perhaps your **DATA _NULL_** began as a paper report many moons ago (if you are as old as the authors, you will recall that is the only kind of report we used to produce). However, since our user community requires their reports in Excel, you now open the .txt file in Excel, define the columnar input and tweak the fonts, colors, etc. to improve the data presentation. Perhaps you even wrote an Excel macro to carry out some of the post processing.

If you recognize this scenario, then it's likely you now have a multi-step process in place that runs the **DATA _NULL_** step, opens the file in Excel, runs an Excel macro, saves the file in Excel format, and publishes. Let's look at a simple example of the *legacy code* and then examine some additions to enhance the Excel output *directly* from SAS and hopefully avoid most, if not all, of the post processing.

EXAMPLE 1A: LEGACY CODE

Consider this simple data step to create a listing report. You had to figure out proper columns for column headings and data, so that everything lined up nicely, which meant anticipating *future data* as well. Your legacy code likely has a lot of computations, formatting, and complexity but for illustrative purposes, our focus is simply on the output.

```
filename out 'c:\temp\HOW_1a_legacy.txt';
data _null_;
  file out;
  if _n_ = 1 then             /*title and Column Hdgs*/
      put  @8 'Legacy Code - Text File Output' //
           @1 'Name' @20 'Gender' @30 'Age' @40 'Height';
```

```
    set sashelp.class;
    format height 5.1;
    put @1 name @23 sex @30 age @40 height;
  run;
```

Partial Output – TXT file contents

```
       Legacy Code - Text File Output

Name                  Gender     Age        Height
Alfred                   M       14         69.0
Alice                    F       13         56.5
Barbara                  F       13         65.3
Carol                    F       14         62.8
Henry                    M       14         63.5
James                    M       12         57.3
Jane                     F       12         59.8
Janet                    F       15         62.5
Jeffrey                  M       13         62.5
```

EXAMPLE 1B: LEGACY CODE – MODIFIED FOR SIMPLE EXCEL OUTPUT

But, there's an easier way! To output more simply to an XLS file, insert a TAB character ('09'x on ASCII systems; '05'x on EBCDIC) between the data elements to create a tab delimited file. No more worries about *what columns to write to* and no column number alterations needed when the data elements grow in the future.

```
  filename out 'c:\temp\HOW_1b_legacy_tabs.xls';
  data _null_;
    retain t '09'x;                  ** ASCII tab character ('05'x on EBCDIC);
    file out;
    if _n_ = 1 then                  /*title and Column Hdgs*/
        put 'Legacy Code - Tab delimited output' /
            'Name' t 'Gender' t 'Age' t 'Height';
    set sashelp.class;
    put name t sex t age t height;
  run;
```

Partial Output - appearance of file when opened in Excel

```
Legacy Code - Tab delimited output
Name        Gender      Age   Height
Alfred      M            14       69
Alice       F            13     56.5
Barbara     F            13     65.3
Carol       F            14     62.8
Henry       M            14     63.5
James       M            12     57.3
Jane        F            12     59.8
Janet       F            15     62.5
Jeffrey     M            13     62.5
John        M            12       59
```

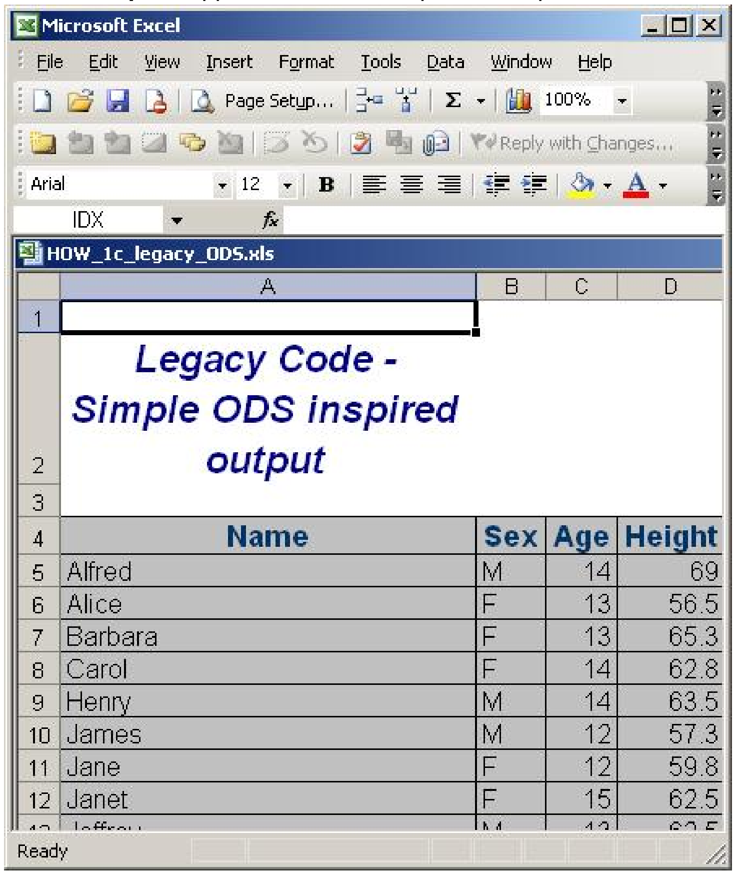
EXAMPLE 1C: ENHANCING LEGACY CODE TO SEND IMPROVED OUTPUT TO EXCEL
In the previous example, titles and column headings were hard-coded using PUT statements. In reality, you likely used multi-row column headings and thus had to deal with getting everything in your code to match up. Furthermore, the Excel file that resulted from the first example had zero formatting features. We got the report into Excel and stopped there!

A few minor alterations to the legacy code produce a slightly more pleasing look:

- remove the manual "tabbing"
- remove the manual titles and column headings, and replace with Title / Footnote statements
- replace the file statement with an ODS statement
- alter the FILE destination.

```
ods listing close;
ods html3 body = 'c:\temp\HOW_1c_legacy_ODS.xls';  * use HTML in v8;
ods noresults;                                     * disables Results Viewer
window;
title "Legacy Code - Simple ODS inspired output";
data _null_;
  set sashelp.class;
  file print
    ods=(var=(name sex age height));
  put name sex age height _ods_;
run;
ods html3 close;
ods listing;
ods results;                                       * enables Results Viewer window;
```

Partial Output - appearance when output file is opened in Excel



Legacy Code - Simple ODS inspired output

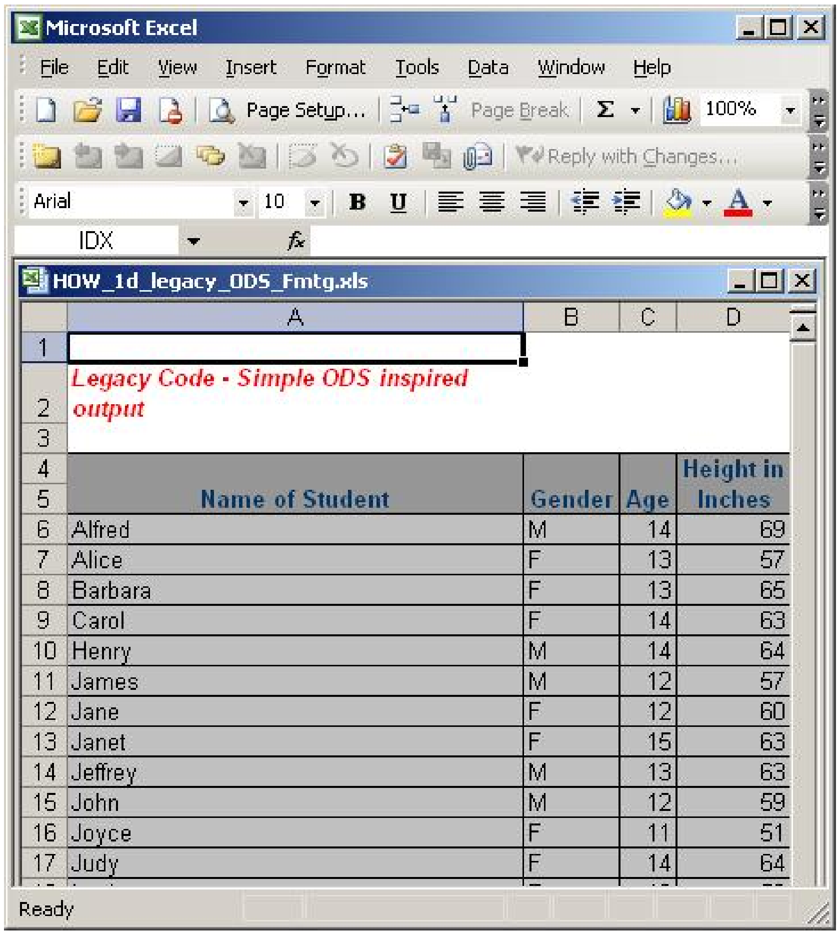
Name	Sex	Age	Height
Alfred	M	14	69
Alice	F	13	56.5
Barbara	F	13	65.3
Carol	F	14	62.8
Henry	M	14	63.5
James	M	12	57.3
Jane	F	12	59.8
Janet	F	15	62.5

EXAMPLE 1D: FURTHER ENHANCEMENTS

There are many easy adjustments that can be made to the program to enhance the output further. ODS automatically takes advantage of any variable labels and formats that are in place. Additionally, you can use the predefined ODS styles to change the Excel appearance. Note the use of the title options to alter the color and justification of the title text. Additional options are available to alter other characteristics such as font type and font size. These same options can be applied to SAS footnotes.

```
ods listing close;
ods html3 body = 'c:\temp\HOW_1d_legacy_ODS_Fmtg.xls' style = statdoc;
ods noresults;
title c=red j=l "Legacy Code - Simple ODS inspired output";
data _null_;
  set sashelp.class;
  label name   = 'Name of Student'
        sex    = 'Gender'
      height   = 'Height in Inches'  ;
  format height 3.;
  file print ods=(var=(name sex age height));
  put name sex age height _ods_;
run;
ods html3 close;
ods listing;
ods results;
```

Partial Output - appearance when output file is opened in Excel


Microsoft Excel
File Edit View Insert Format Tools Data Window Help
Page Setup... Page Break 100%
Reply with Changes...
Arial 10
IDX
HOW_1d_legacy_ODS_Fmtg.xls
Ready

Legacy Code - Simple ODS inspired output

Name of Student	Gender	Age	Height in Inches
Alfred	M	14	69
Alice	F	13	57
Barbara	F	13	65
Carol	F	14	63
Henry	M	14	64
James	M	12	57
Jane	F	12	60
Janet	F	15	63
Jeffrey	M	13	63
John	M	12	59
Joyce	F	11	51
Judy	F	14	64

EXAMPLE 1E: PRODUCING MULTISHEET EXCEL REPORTS

Once you begin publishing results in Excel, it isn't long before you *must have* multi-sheet output! In Version 8, this could be accomplished, but not easily, and it often meant post-processing of the output file with Excel macros. With SAS 9, there are various approaches for producing multi-sheet Excel output. The approach demonstrated here takes advantage of a simple PROC PRINT with a BY statement. For regular output, the BY statement produces sections of output. The **ExcelXP tagset** (SAS 9) produces one worksheet per by group. This functionality requires Excel 2002 or 2003.

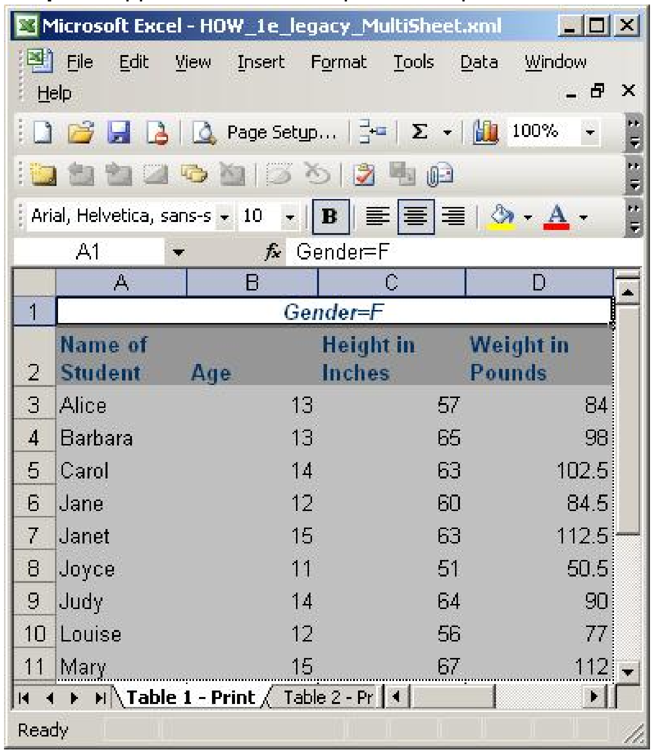
```
ods listing close;
ods tagsets.ExcelXP path='c:\temp' file='HOW_1e_legacy_MultiSheet.xls'
    style=statdoc options(doc='help' sheet_interval='bygroup' );
     /* doc='help' lists all ExcelXP tagset options, in the LOG */
ods noresults;

title "Re-worked Legacy Code - Multi-Sheet Excel Workbook";
proc sort data=sashelp.class
          out=class;
   by sex;
run;

proc print data=class noobs label split='*';
  by sex;
  label name   = 'Name of*Student'
        sex    = 'Gender'
        age    = '*Age'
        weight = 'Weight in*Pounds'
        height = 'Height in*Inches';
  format height 3.;
run;
ods tagsets.ExcelXP close;
ods listing;
ods results;
```

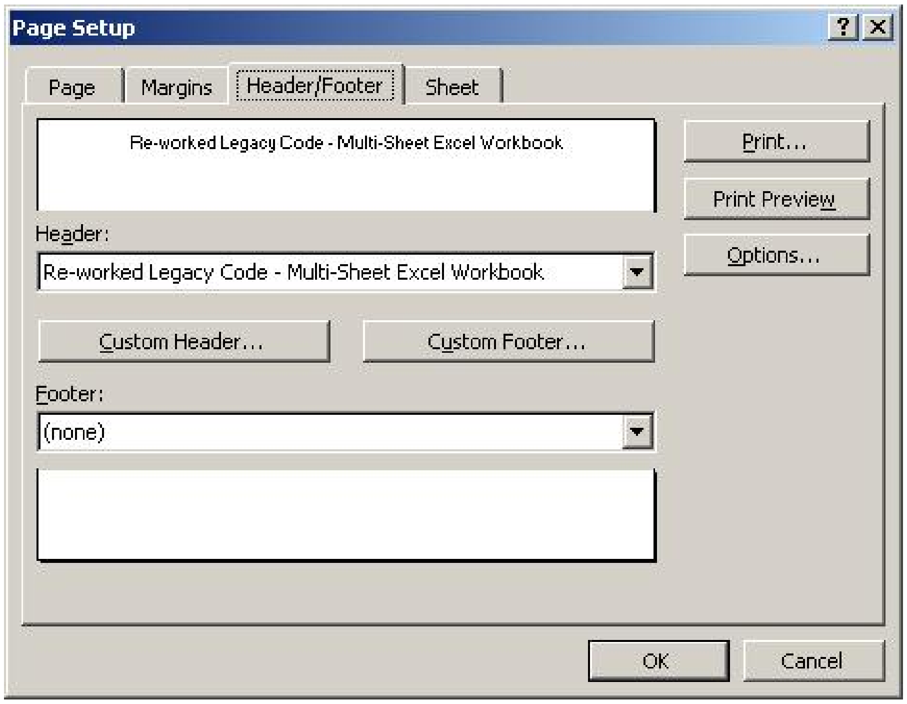
When the output file is opened in Excel, notice that there are two sheets with naming as produced by ODS. And, the title is in the Header of the worksheet!

Output – appearance when output file is opened in Excel


Microsoft Excel - HOW_1e_legacy_MultiSheet.xml
File Edit View Insert Format Tools Data Window Help
Page Setup...
100%
Arial, Helvetica, sans-s
10
A1
Gender=F
Table 1 - Print
Table 2 - Pr
Ready

	A	B	C	D
1	*Gender=F*			
2	**Name of Student**	**Age**	**Height in Inches**	**Weight in Pounds**
3	Alice	13	57	84
4	Barbara	13	65	98
5	Carol	14	63	102.5
6	Jane	12	60	84.5
7	Janet	15	63	112.5
8	Joyce	11	51	50.5
9	Judy	14	64	90
10	Louise	12	56	77
11	Mary	15	67	112

The "Page Setup" window shows the Excel heading created by the SAS title statement:



Re-worked Legacy Code - Multi-Sheet Excel Workbook

Header: Re-worked Legacy Code - Multi-Sheet Excel Workbook

Footer: (none)

EXAMPLE 1F: CONTROLLING EXCEL FEATURES FROM SAS

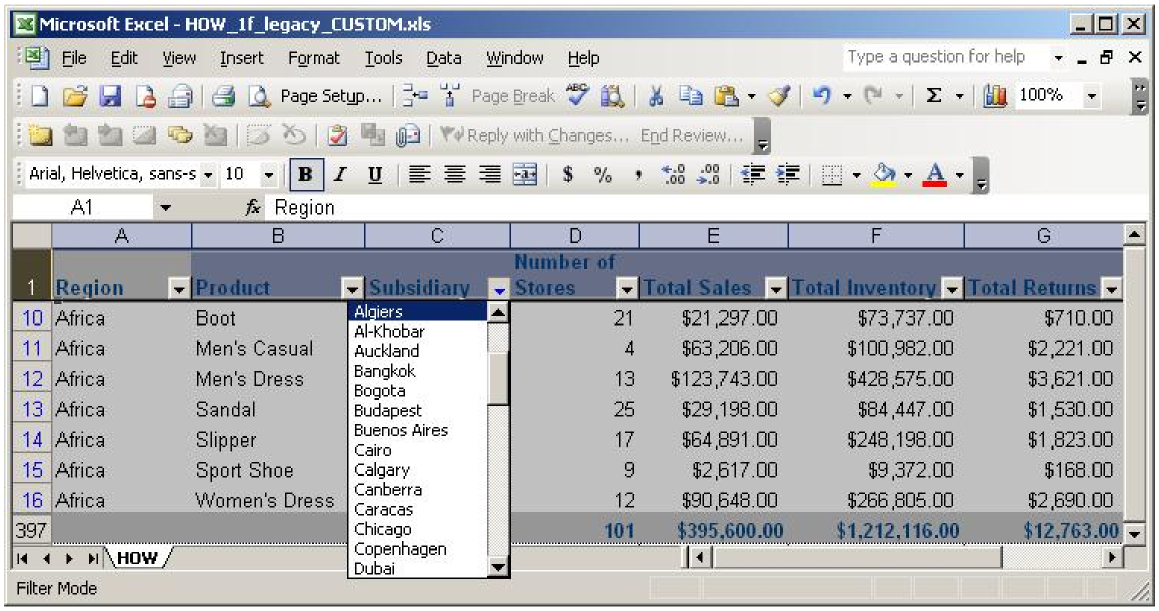
Instead of post-processing the work book to modify sheet names, and change functionality such as headers and data filters, the ExcelXP tagset has *many* options that enable further worksheet control. Excel 2002 or 2003 required.

```
ods listing close;
ods tagsets.ExcelXP path='c:\temp' file='HOW_1f_legacy_CUSTOM.xls' style=statdoc
    options (sheet_name = 'SESUG_HOW' frozen_headers = 'Yes' autofilter = 'All'
             auto_subtotals = 'Yes' );
ods noresults;

title "Controlling Excel Features via ODS";
proc print data=sashelp.shoes noobs;
  sum stores sales inventory returns;
run;

ods tagsets.ExcelXP close;
ods listing;
ods results;
```

Output – as it appears when you open the file in Excel. Subsidiary "Algiers" has been selected using the auto-filter, notice the **automatic sub-totals** in line 397.


Microsoft Excel - HOW_1f_legacy_CUSTOM.xls

A1 fx Region

	A	B	C	D	E	F	G
1	Region	Product	Subsidiary	Number of Stores	Total Sales	Total Inventory	Total Returns
10	Africa	Boot		21	$21,297.00	$73,737.00	$710.00
11	Africa	Men's Casual		4	$63,206.00	$100,982.00	$2,221.00
12	Africa	Men's Dress		13	$123,743.00	$428,575.00	$3,621.00
13	Africa	Sandal		25	$29,198.00	$84,447.00	$1,530.00
14	Africa	Slipper		17	$64,891.00	$248,198.00	$1,823.00
15	Africa	Sport Shoe		9	$2,617.00	$9,372.00	$168.00
16	Africa	Women's Dress		12	$90,648.00	$266,805.00	$2,690.00
397				101	$395,600.00	$1,212,116.00	$12,763.00

Using the SAS 9 ExcelXP tagset, there is a wealth of **additional features** to enhance and control Excel output. See the Recommended Reading section of this paper for a number of excellent ExcelXP tagset resources.

EXCEL PIVOT TABLES – POPULATING THEM PAINLESSLY

After the users start working with your Excel reports, it will not be long before they ask for further functionality including "slice and dice" capability. Excel **pivot tables** enable you to explore many views of the data including high level summaries while providing drill down to more detailed layers. Typically, an Excel Pivot Table has dimensions (row, column and page variables) along with fact data (such as Total Sales, Average Volume, and Highest Temperature). Excel pivot tables are typically created using the Excel PivotTable wizard. In addition, due to Excel's limitation on maximum rows and the ease with which external data can be refreshed, pivot tables are commonly populated from an external data source.

When you publish your pivot table report, you must have a strategy for ***easily* refreshing the data**. This also means you need a reliable and consistent mechanism for feeding source data into Excel. Two candidates are **CSV** and **XML** files, both of which are easily produced by SAS. Unless drill down capability to detail data is required, pivot tables work **best** if they're sourced from a summarized ***cube*** rather than a large detail file. PROC summary is great at producing cubes.

The sample data used in the following example is from the summarized data in SASHELP.PRDSALE:

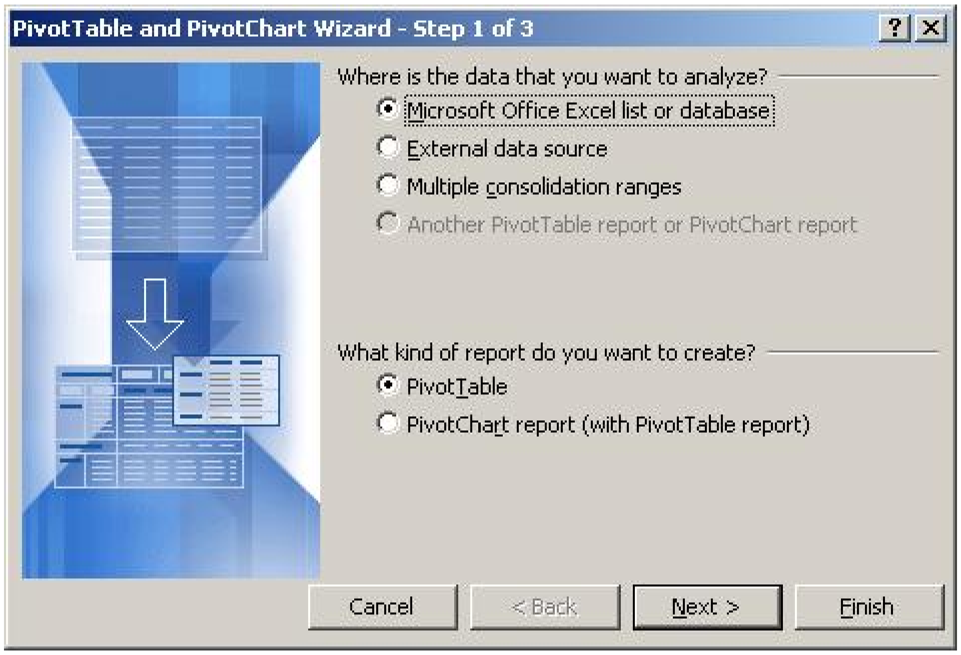
```
proc export data=sashelp.prdsale
            outfile='c:\temp\PrdSale_dataInExcel.xls';
run;
```

After opening c:\temp\PrdSale_dataInExcel.xls:

A SIMPLE PIVOT TABLE

To create a pivot table using data that already exists in Excel,

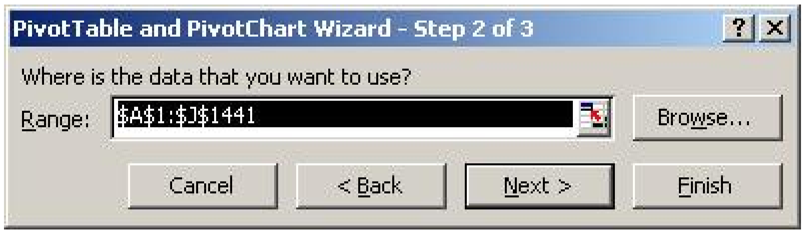
- Open the excel spreadsheet that contains your data
- Select **Data ➔ Pivot Table and Pivot Chart Report**
- Notice that the data source defaults to Microsoft Office Excel list or database, which is correct in this instance



Where is the data that you want to analyze?
[✓ Microsoft Office Excel list or database]
External data source
Multiple consolidation ranges
Another PivotTable report or PivotChart report

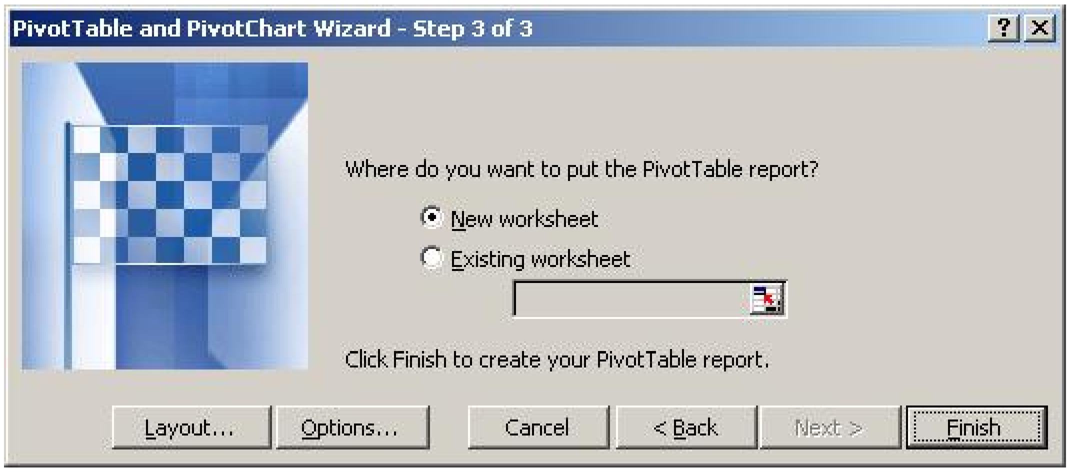
What kind of report do you want to create?
[✓ PivotTable]
PivotChart report (with PivotTable report)

- Click **Next**
- By default, Excel assumes that all of the rows and columns are included. Note that Excel understands that the column names are in row 1. Confirm by clicking **Next**.



Where is the data that you want to use?
Range: $A$1:$J$1441

- Accept the default of **New Worksheet** for locating the data source (note that you can then hide the original data so it is not visible to the report consumers).



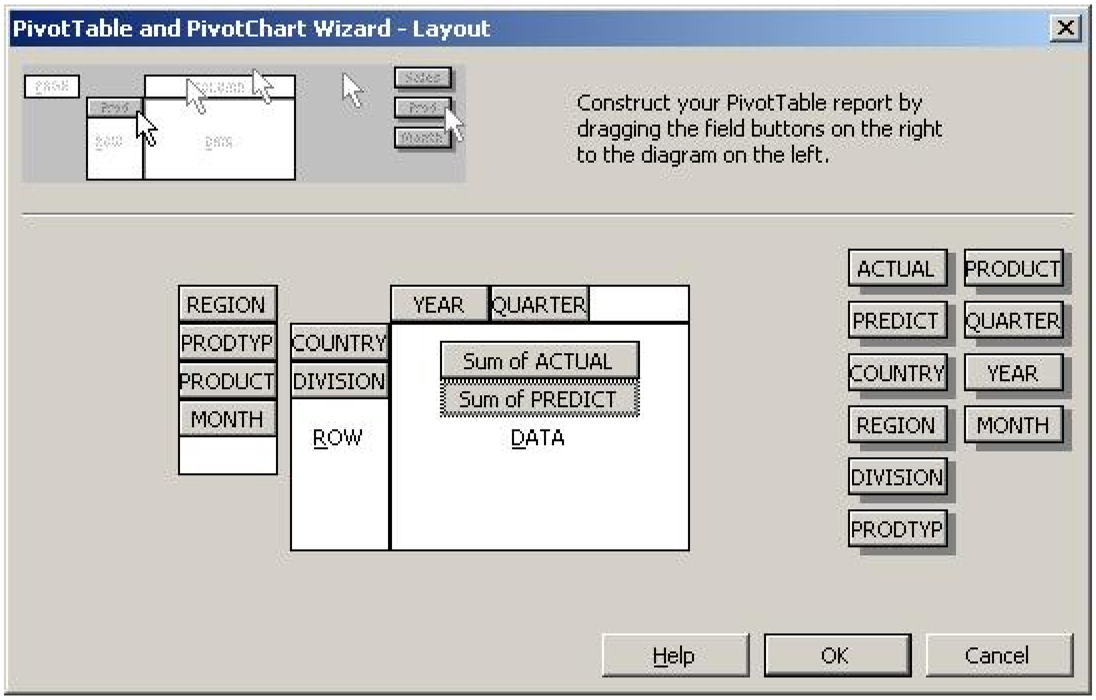
PivotTable and PivotChart Wizard - Step 3 of 3

Where do you want to put the PivotTable report?

● New worksheet
○ Existing worksheet

Click Finish to create your PivotTable report.

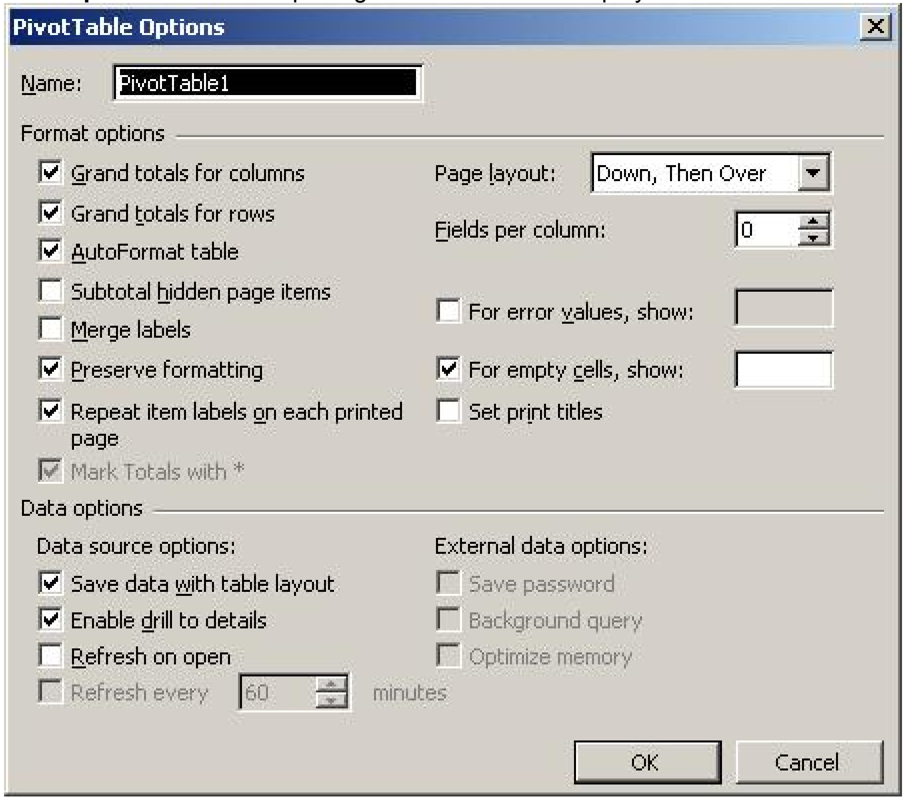
- Click **Layout** to design the pivot and move the columns into their initial roles as shown. Note that the report user can change the roles of all columns to meet their reporting needs. Click **OK**.



PivotTable and PivotChart Wizard - Layout

Construct your PivotTable report by dragging the field buttons on the right to the diagram on the left.

- Click **Options** to control reporting features and data display. Click **OK**.



PivotTable Options

Name: PivotTable1

Format options

- [✓] Grand totals for columns
- [✓] Grand totals for rows
- [✓] AutoFormat table
- [ ] Subtotal hidden page items
- [ ] Merge labels
- [✓] Preserve formatting
- [✓] Repeat item labels on each printed page
- [✓] Mark Totals with *

Page layout: Down, Then Over

Fields per column: 0

- [ ] For error values, show:
- [✓] For empty cells, show:
- [ ] Set print titles

Data options

Data source options:

- [✓] Save data with table layout
- [✓] Enable drill to details
- [ ] Refresh on open
- [ ] Refresh every 60 minutes

External data options:

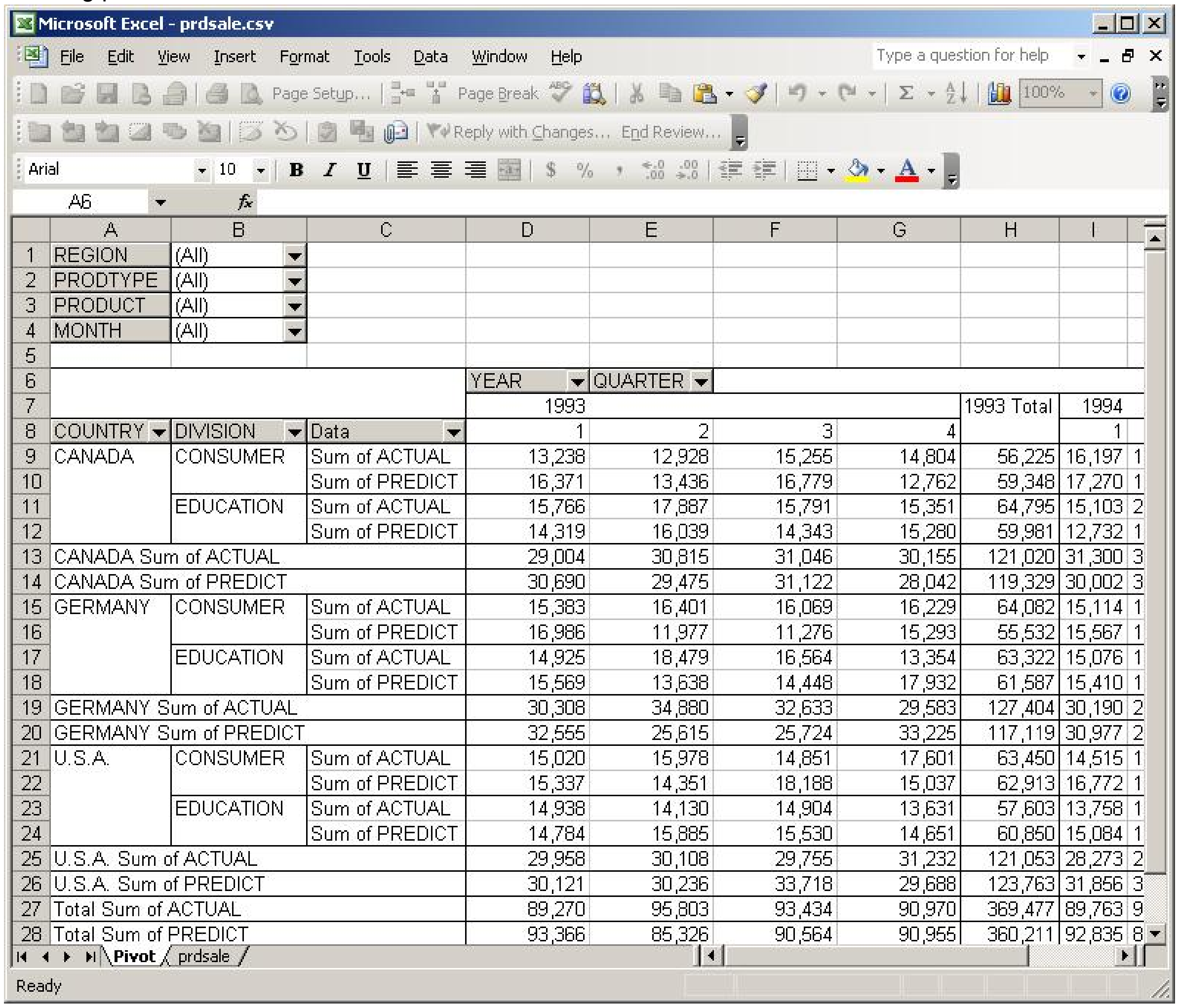
- [ ] Save password
- [ ] Background query
- [ ] Optimize memory

OK Cancel

- Note the "**Refresh on open**" check-box on the bottom left of the options panel. Keep this in mind when we discuss CSV sourced pivot tables in a later section of the paper!

- Click **OK** and then **Finish** to complete the design phase and display the Pivot Table report.

The initial display of the pivot table may not have optimum formatting and column widths. However, these can be changed easily through **field settings** and other options. Following a few field setting and width changes, the resulting pivot table is:


Microsoft Excel - prdsale.csv
File Edit View Insert Format Tools Data Window Help
Type a question for help
Page Setup... Page Break
Reply with Changes... End Review...
Arial 10
100%
A6
Pivot prdsale
Ready

	A	B	C	D	E	F	G	H	I	
1	REGION	(All)								
2	PRODTYPE	(All)								
3	PRODUCT	(All)								
4	MONTH	(All)								
5										
6				YEAR	QUARTER					
7				1993				1993 Total	1994	
8	COUNTRY	DIVISION	Data	1	2	3	4		1	
9	CANADA	CONSUMER	Sum of ACTUAL	13,238	12,928	15,255	14,804	56,225	16,197	1
10			Sum of PREDICT	16,371	13,436	16,779	12,762	59,348	17,270	1
11		EDUCATION	Sum of ACTUAL	15,766	17,887	15,791	15,351	64,795	15,103	2
12			Sum of PREDICT	14,319	16,039	14,343	15,280	59,981	12,732	1
13	CANADA Sum of ACTUAL			29,004	30,815	31,046	30,155	121,020	31,300	3
14	CANADA Sum of PREDICT			30,690	29,475	31,122	28,042	119,329	30,002	3
15	GERMANY	CONSUMER	Sum of ACTUAL	15,383	16,401	16,069	16,229	64,082	15,114	1
16			Sum of PREDICT	16,986	11,977	11,276	15,293	55,532	15,567	1
17		EDUCATION	Sum of ACTUAL	14,925	18,479	16,564	13,354	63,322	15,076	1
18			Sum of PREDICT	15,569	13,638	14,448	17,932	61,587	15,410	1
19	GERMANY Sum of ACTUAL			30,308	34,880	32,633	29,583	127,404	30,190	2
20	GERMANY Sum of PREDICT			32,555	25,615	25,724	33,225	117,119	30,977	2
21	U.S.A.	CONSUMER	Sum of ACTUAL	15,020	15,978	14,851	17,601	63,450	14,515	1
22			Sum of PREDICT	15,337	14,351	18,188	15,037	62,913	16,772	1
23		EDUCATION	Sum of ACTUAL	14,938	14,130	14,904	13,631	57,603	13,758	1
24			Sum of PREDICT	14,784	15,885	15,530	14,651	60,850	15,084	1
25	U.S.A. Sum of ACTUAL			29,958	30,108	29,755	31,232	121,053	28,273	2
26	U.S.A. Sum of PREDICT			30,121	30,236	33,718	29,688	123,763	31,856	3
27	Total Sum of ACTUAL			89,270	95,803	93,434	90,970	369,477	89,763	9
28	Total Sum of PREDICT			93,366	85,326	90,564	90,955	360,211	92,835	8

The pivot table we just created is great for navigating through the data and exploring trends. But, sourcing the pivot table *directly* from Excel is not ideal if

- your detail data (or cube) exceeds Excel's row limit
- you plan on regular data updates.

A more production-oriented approach is to source the Pivot Table from an external data source, so that whenever the Pivot opens, the table can be refreshed (synchronized) with the current data. A pivot table can be sourced from a wide variety of external data including XML, CSV files, MS Access data bases, and many others.

CREATING XML DATA FROM SAS – EASILY

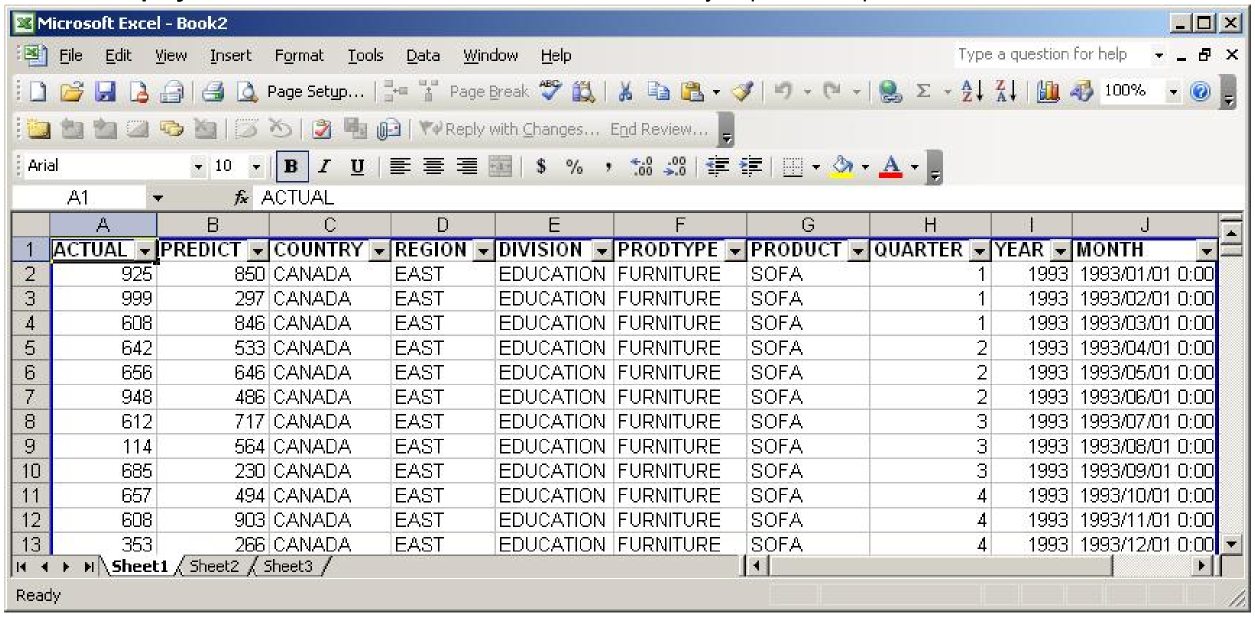
Since version 8, SAS has been able to read and write XML data. In SAS 9 this capability has been greatly enhanced. While the nitty-gritty details of XML representation, XML maps and the like are beyond the scope of this paper, the following simple example illustrates how easy it is to create XML data files.

Without knowing anything about XML file specifics, you can easily create an XML file from your summary data. In this example, we show a typical summary step to produce a cube from detail data, for instructional purposes. Please note that sashelp.prdsale is *already summarized* so a simple copy to the SESUG libref is all that would *really* be needed.

```
libname sesug xml 'C:\temp\Prdsale_xml.xml'; ** available in V8 and SAS 9;
/* these data are already summarized - code illustrates "creating the cube" */
proc summary data=sashelp.prdsale nway;
   class country region division prodtype product year quarter month;
   var actual predict;
   output out=sesug.pivot(drop = _type_ _freq_) sum=;
run;
```

If you open the resulting file **with Excel**, and when prompted, select XML List as display format, the initial view is:

XML file displayed in Excel 2002/2003 – filters are automatically in place on open



	A	B	C	D	E	F	G	H	I	J
1	ACTUAL	PREDICT	COUNTRY	REGION	DIVISION	PRODTYPE	PRODUCT	QUARTER	YEAR	MONTH
2	925	850	CANADA	EAST	EDUCATION	FURNITURE	SOFA	1	1993	1993/01/01 0:00
3	999	297	CANADA	EAST	EDUCATION	FURNITURE	SOFA	1	1993	1993/02/01 0:00
4	608	846	CANADA	EAST	EDUCATION	FURNITURE	SOFA	1	1993	1993/03/01 0:00
5	642	533	CANADA	EAST	EDUCATION	FURNITURE	SOFA	2	1993	1993/04/01 0:00
6	656	646	CANADA	EAST	EDUCATION	FURNITURE	SOFA	2	1993	1993/05/01 0:00
7	948	486	CANADA	EAST	EDUCATION	FURNITURE	SOFA	2	1993	1993/06/01 0:00
8	612	717	CANADA	EAST	EDUCATION	FURNITURE	SOFA	3	1993	1993/07/01 0:00
9	114	564	CANADA	EAST	EDUCATION	FURNITURE	SOFA	3	1993	1993/08/01 0:00
10	685	230	CANADA	EAST	EDUCATION	FURNITURE	SOFA	3	1993	1993/09/01 0:00
11	657	494	CANADA	EAST	EDUCATION	FURNITURE	SOFA	4	1993	1993/10/01 0:00
12	608	903	CANADA	EAST	EDUCATION	FURNITURE	SOFA	4	1993	1993/11/01 0:00
13	353	266	CANADA	EAST	EDUCATION	FURNITURE	SOFA	4	1993	1993/12/01 0:00

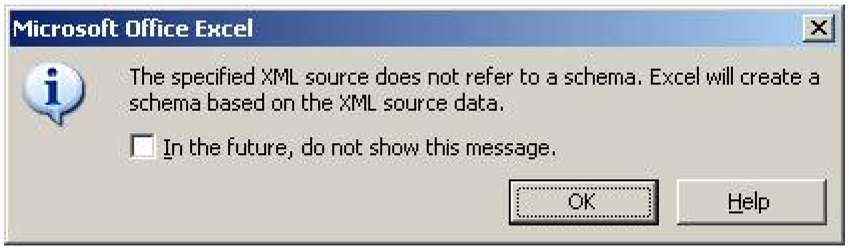
While this demonstrates ***one more way to get data from SAS into Excel – easily***, our goal for creating the XML file is to source a ***refreshable*** pivot table using the XML data. Unfortunately, it won't be as seamless as we might hope.

PIVOT TABLE - DATA SOURCE = XML

If you have production code that cranks out regular updates to your data in XML format, you can source an Excel Pivot Table with XML.

Open a new Excel workbook.

- Select **Data → XML → Import …**
- Locate the XML file to import and click **Import**
- Click **OK** from the dialog asking about creating a Schema

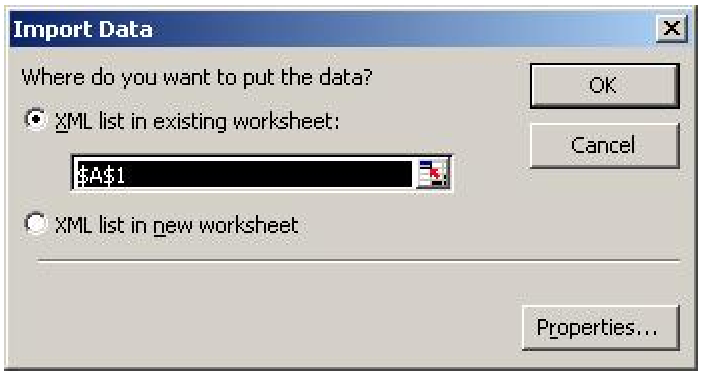


Microsoft Office Excel

The specified XML source does not refer to a schema. Excel will create a schema based on the XML source data.

☐ In the future, do not show this message.

- Click **OK** to confirm the placement of the imported data (begin in first cell)


Import Data
Where do you want to put the data?
XML list in existing worksheet:
$A$1
XML list in new worksheet
OK
Cancel
Properties...

The data now reside in Excel. Notice that a **refresh XML data** choice is available!

Mouse button 2 ➔ **XML** ➔ **Refresh XML data**.

To build a pivot table from these data, you simply follow the identical steps used to create the pivot table in an earlier section. To ensure that the users are always accessing the most current data, a **double**-refresh is required!! You must go into the XML-sourced sheet, right-click ➔ **XML** ➔ **Refresh XML data** before opening the pivot**.** It's also necessary to ensure the table option for **refresh on open** is checked so the latest XML data are loaded when the pivot table displays. The requirement for the double-refresh makes **XML a poor choice for sourcing your pivot table**. The next section of the paper outlines the use of CSV files, which are currently a *much* better choice than XML for creating automatically updating pivot tables.

DEFINING THE DATA SOURCE - CSV

Populating your pivot table from XML is easy. However, if you must load data of more than Excel's 64K row limit or you cannot deal with the REALLY huge XML file (bloated by presence of XML tags), an alternative exists! Instead of sourcing your pivot table through XML, you can source it from a CSV (comma separated value) file which is **not limited** by Excel's 64K row limit. However, since **CSV is not one of the default data souces in the Pivot table wizard**, a new data source must be defined.

Step 1: You can easily create a CSV from your SAS data set using ODS:

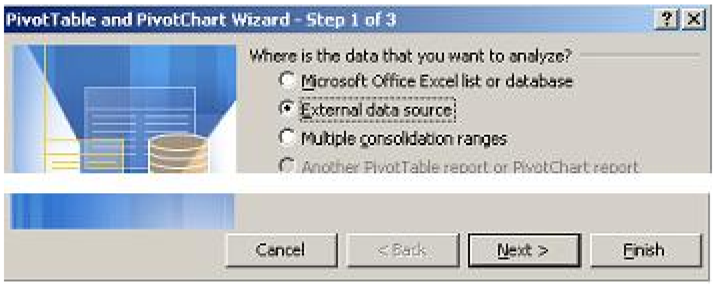
```
ods noresults;
ods listing close;
ods csv file='c:\temp\prdsale.csv';

proc print data=sashelp.prdsale noobs;
run;

ods csv close;
ods listing;
ods results;
```

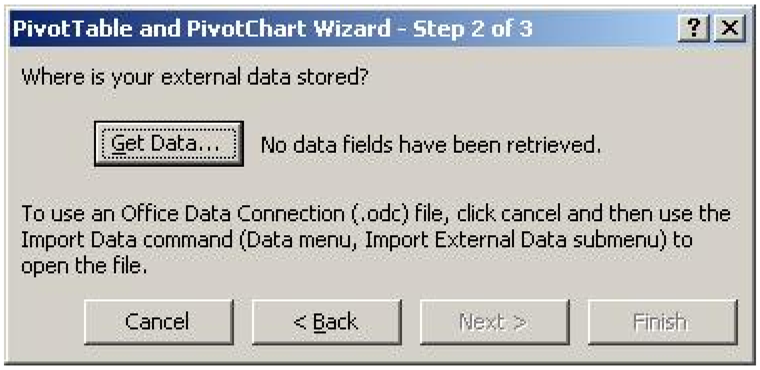
Step 2: The PivotTable Wizard will guide us through the maze of external data sources, to create a new data source:

- Select **Data** ➔ **PivotTable and PivotChart Report …**
- Select **External Data Source** and click **Next**


PivotTable and PivotChart Wizard - Step 1 of 3

Where is the data that you want to analyze?

( ) Microsoft Office Excel list or database
[✓ External data source]
( ) Multiple consolidation ranges
( ) Another PivotTable report or PivotChart report

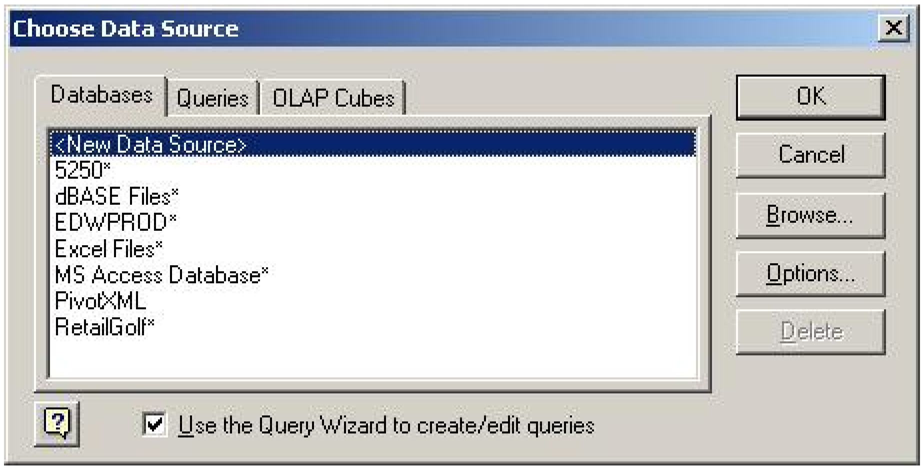

PivotTable and PivotChart Wizard - Step 2 of 3

Where is your external data stored?

No data fields have been retrieved.

To use an Office Data Connection (.odc) file, click cancel and then use the Import Data command (Data menu, Import External Data submenu) to open the file.

- click **Get Data**

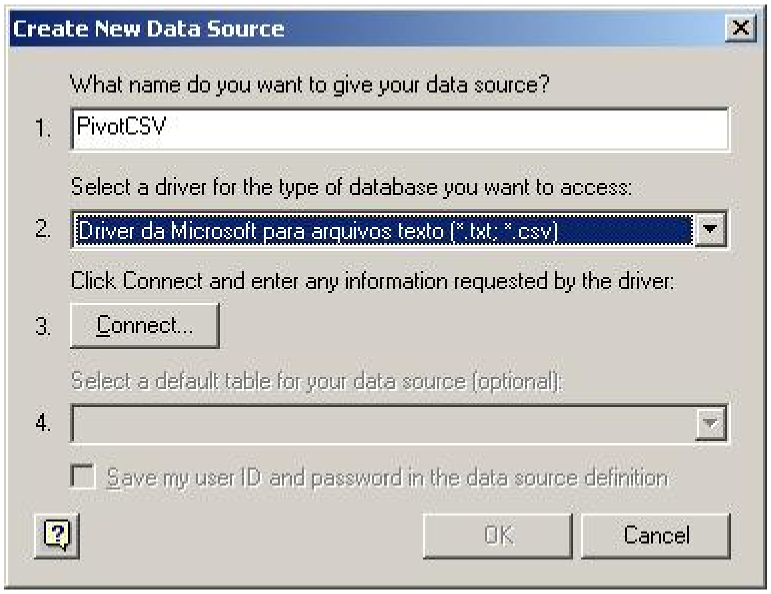

Choose Data Source

Databases | Queries | OLAP Cubes

<New Data Source>
5250*
dBASE Files*
EDWPROD*
Excel Files*
MS Access Database*
PivotXML
RetailGolf*

[✓] Use the Query Wizard to create/edit queries

- Select **<New Data Source>** and click **OK**


Create New Data Source

What name do you want to give your data source?

1. PivotCSV

Select a driver for the type of database you want to access:

2. Driver da Microsoft para arquivos texto (*.txt; *.csv)

Click Connect and enter any information requested by the driver:

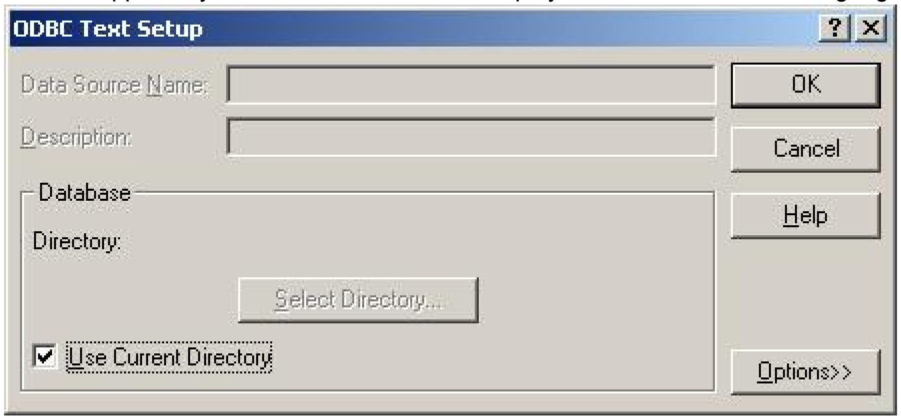
3. Connect...

Select a default table for your data source (optional):

4.

☐ Save my user ID and password in the data source definition

- Enter a name for the data source, select the **CSV driver**, and click **Connect**.
  It is apparently a *feature* of Microsoft to display the drivers in another language ☺


ODBC Text Setup

Data Source Name:

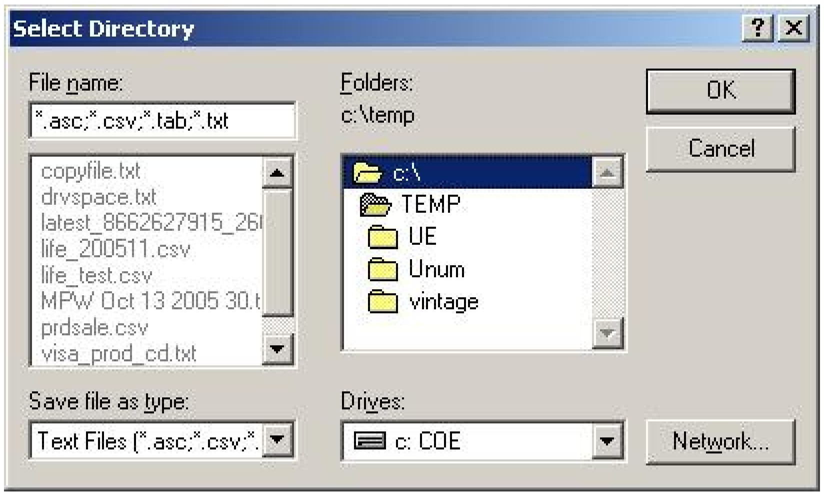
Description:

Database

Directory:

☑ Use Current Directory

- Uncheck **Use Current Directory** and click **Select Directory** to navigate to the location of your CSV file


Select Directory

File name: *.asc;*.csv;*.tab;*.txt

copyfile.txt
drvspace.txt
latest_8662627915_26
life_200511.csv
life_test.csv
MPW Oct 13 2005 30.t
prdsale.csv
visa_prod_cd.txt

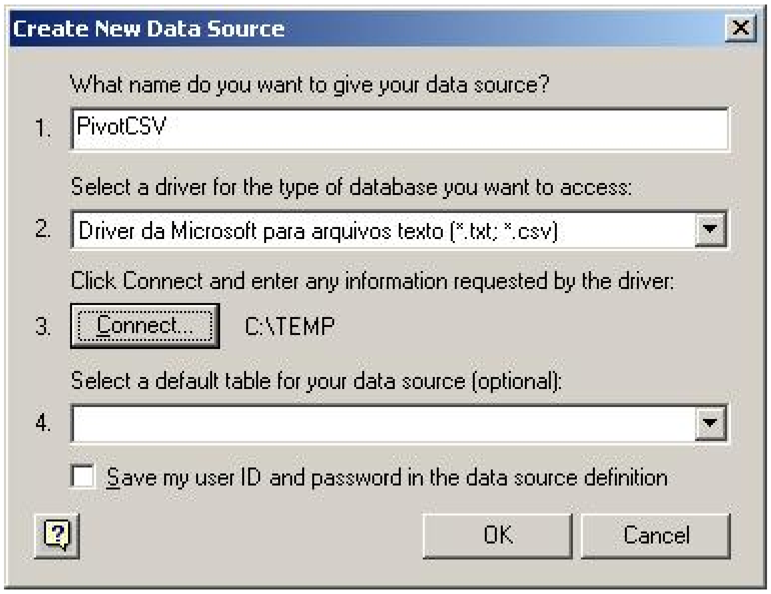
Folders: c:\temp

c:\
TEMP
UE
Unum
vintage

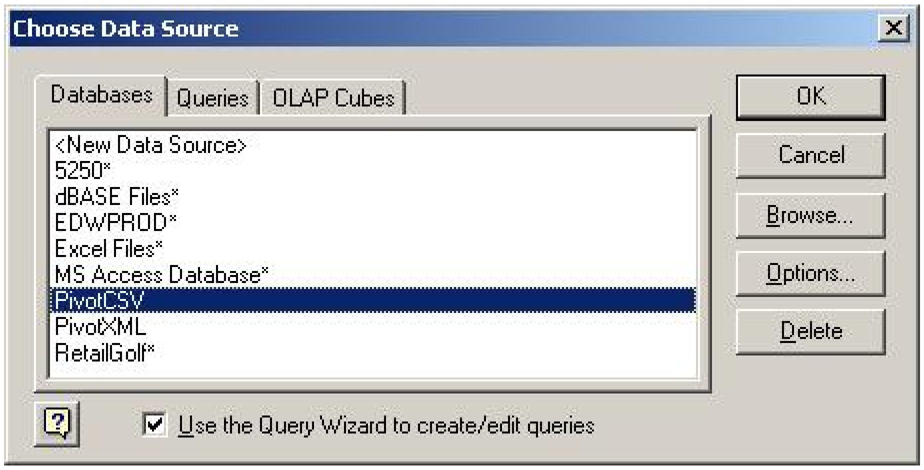
Save file as type: Text Files (*.asc;*.csv;*.

Drives: c: COE

- click **OK** and **OK** again


Create New Data Source
What name do you want to give your data source?
1. PivotCSV
Select a driver for the type of database you want to access:
2. Driver da Microsoft para arquivos texto (*.txt; *.csv)
Click Connect and enter any information requested by the driver:
3. Connect... C:\TEMP
Select a default table for your data source (optional):
4.
Save my user ID and password in the data source definition
OK
Cancel

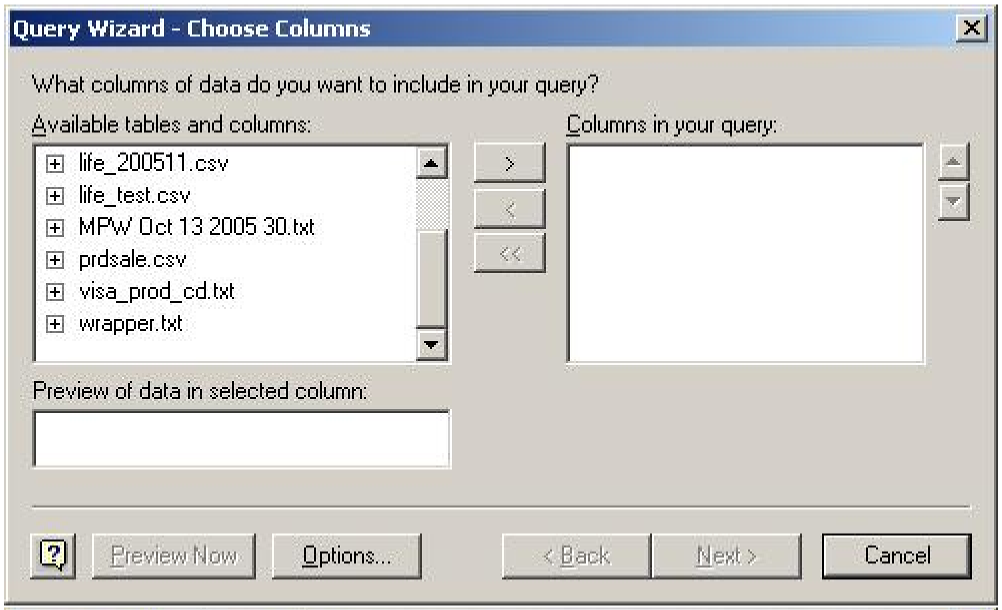
- click **OK**

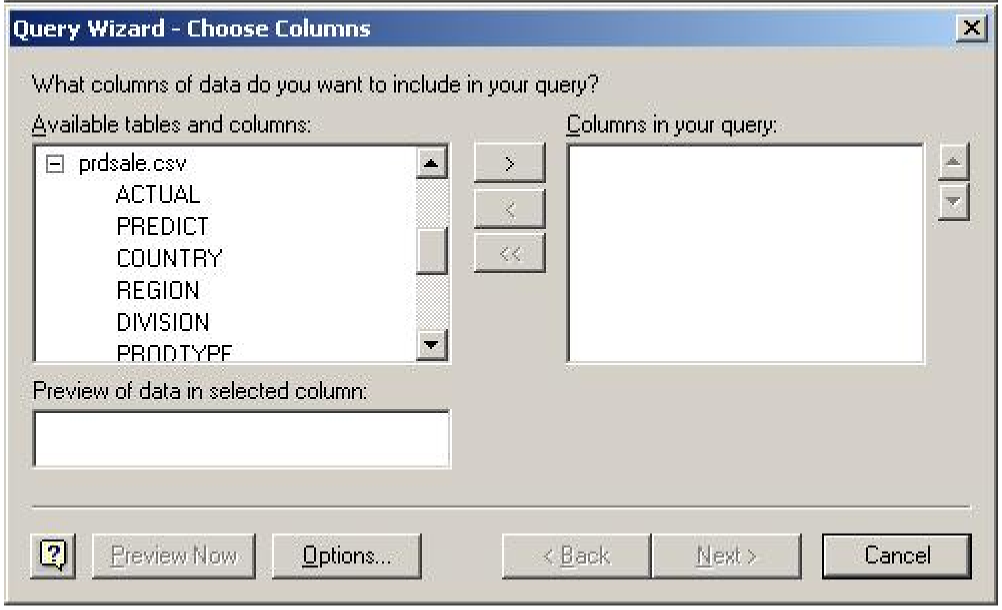


- ensure the data source you just added is high-lighted and click **OK**

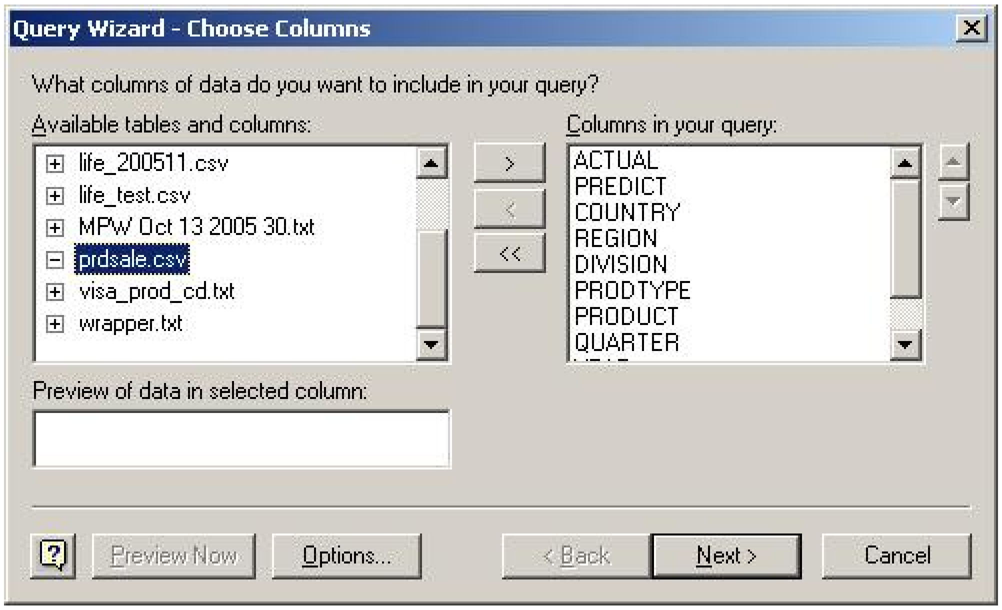
The data source is now defined and will be available to you in Excel!
Now, you can focus on selecting the CSV you would like to use and then assemble the columns and layout you require.

- the CSV file we're looking for is **prdsale.csv**, scroll down until it appears in the list, click the **+** sign to view the list of available fields


Query Wizard - Choose Columns
What columns of data do you want to include in your query?
Available tables and columns:
life_200511.csv
life_test.csv
MPW Oct 13 2005 30.txt
prdsale.csv
visa_prod_cd.txt
wrapper.txt
Columns in your query:
Preview of data in selected column:
Preview Now
Options...
< Back
Next >
Cancel


Query Wizard - Choose Columns
What columns of data do you want to include in your query?
Available tables and columns:
prdsale.csv
ACTUAL
PREDICT
COUNTRY
REGION
DIVISION
PRODTYPE
Columns in your query:
Preview of data in selected column:
Preview Now
Options...
< Back
Next >
Cancel

- double-click on desired columns or select the column name and click **>** to select them. If all columns of **prdsale.csv** are to be selected, click the CSV file name and click **>**.


Query Wizard - Choose Columns

What columns of data do you want to include in your query?

Available tables and columns:

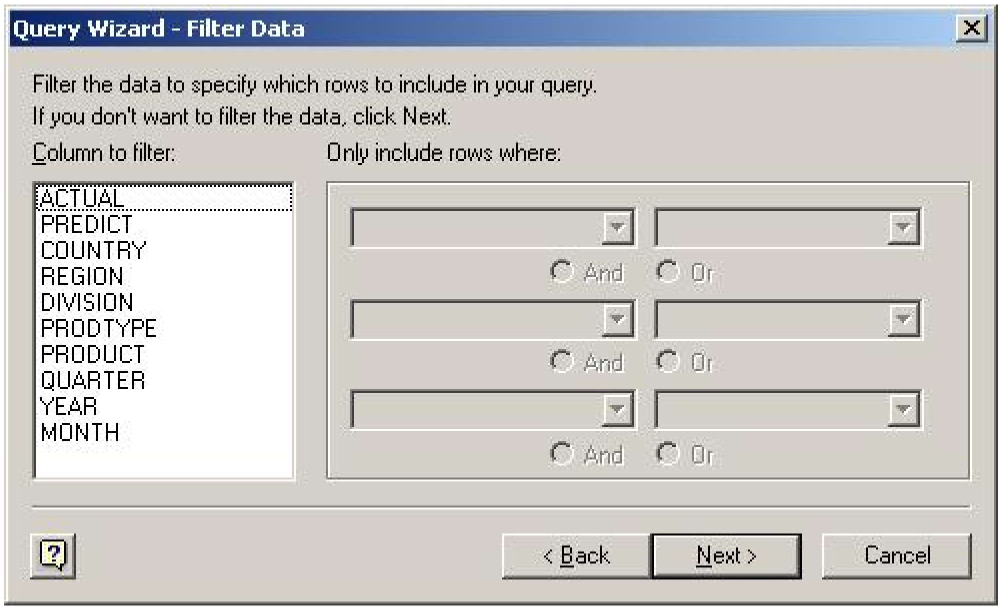
- life_200511.csv
- life_test.csv
- MPW Oct 13 2005 30.txt
- prdsale.csv
- visa_prod_cd.txt
- wrapper.txt

Columns in your query:

ACTUAL
PREDICT
COUNTRY
REGION
DIVISION
PRODTYPE
PRODUCT
QUARTER

Preview of data in selected column:

○ click **Next**


Query Wizard - Filter Data

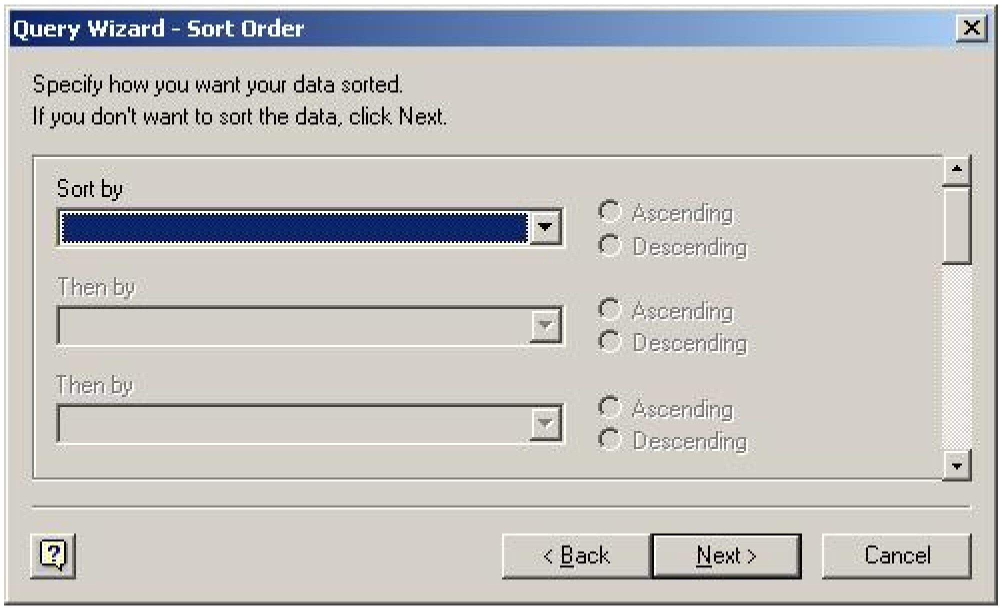
Filter the data to specify which rows to include in your query.
If you don't want to filter the data, click Next.

Column to filter:

ACTUAL
PREDICT
COUNTRY
REGION
DIVISION
PRODTYPE
PRODUCT
QUARTER
YEAR
MONTH

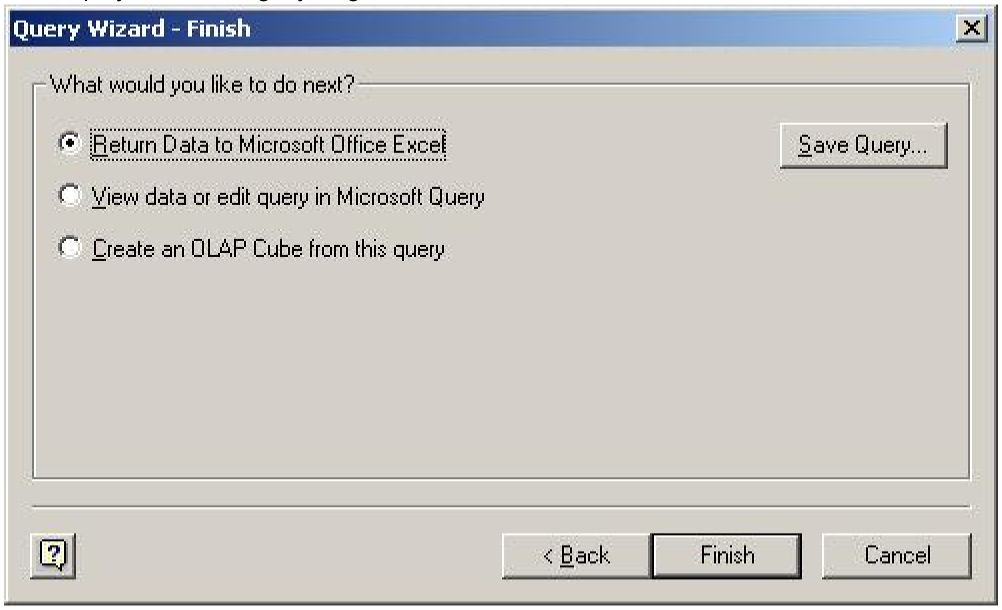
Only include rows where:

○ if filtering is required, select the columns and filtering rules and click **Next**


Query Wizard - Sort Order

Specify how you want your data sorted.
If you don't want to sort the data, click Next.

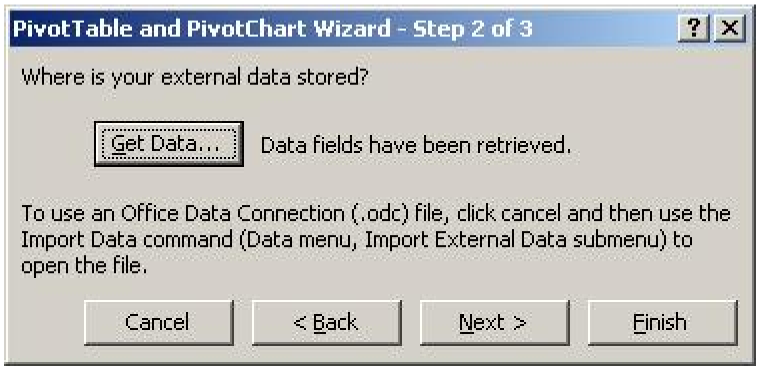
- if required, select sort criteria and click **Next**. If you don't specify SORT criteria, your initial display could be slightly illogical.


Query Wizard - Finish

What would you like to do next?

(●) Return Data to Microsoft Office Excel
( ) View data or edit query in Microsoft Query
( ) Create an OLAP Cube from this query

- select **Return Data to Microsoft Office Excel** and click **Finish** to bring control back to the PivotTable Wizard.


PivotTable and PivotChart Wizard - Step 2 of 3

Where is your external data stored?

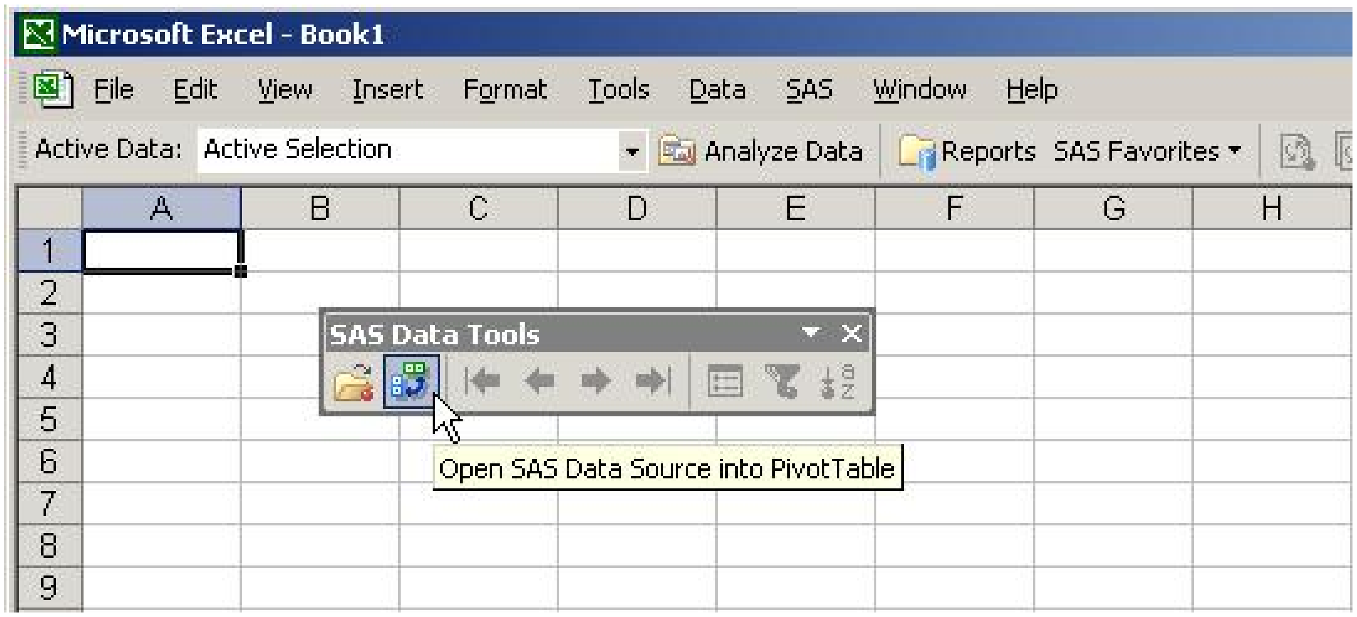
Data fields have been retrieved.

To use an Office Data Connection (.odc) file, click cancel and then use the Import Data command (Data menu, Import External Data submenu) to open the file.

- click **Next** to continue with the PivotTable Wizard (see earlier instructions for field selection and customization)
- Remember to select “Refresh on open” in the PivotTable options panel if this action is required.

BRIGHT HOPE FOR THE FUTURE!

With the advent of SP4 in v9.1, the SAS Add-in for Microsoft Office component of the BI platform provides the ability to source Excel PivotTables *directly* from SAS datasets! No more intermediate CSV files, Access Databases or nasty Data Source setup steps to endure.


Microsoft Excel - Book1
File Edit View Insert Format Tools Data SAS Window Help
Active Data: Active Selection
Analyze Data
Reports SAS Favorites
SAS Data Tools
Open SAS Data Source into PivotTable

CONCLUSION

Whew, this paper covers a lot of ground !! Did you think we’d get from ancient data _null_ listing reports to XML data files feeding publishable Excel OLAP cubes in one paper?

SAS provides the tools to get there from here. Using the still-evolving ExcelXP tagsets in conjunction with customizable style templates, you *can* create Excel spreadsheet output that’s ready for publication with very little, if any, manual intervention. And, if you’re currently producing listing reports, or delimited output that requires you to manually massage your report files, SAS has made it relatively painless to affect the code changes required to seamlessly produce *slick* Excel output.

ACKNOWLEDGMENTS

Eric Gebhart of SAS for his expert assistance with ExcelXP tagset details.

RECOMMENDED READING

DelGobbo, V. 2006. "Creating AND Importing Multi-Sheet Excel Workbooks the Easy Way with SAS®". Proceedings of the Thirty-First Annual SAS Users Group International Conference, 31. CD-ROM. Paper 115.

Gebhart, E. 2005. " ODS Markup: The SAS® Reports You've Always Dreamed Of ". Proceedings of the Thirtieth Annual SAS Users Group International Conference, 30. CD-ROM. Paper 85.

Zender, C. 2005. "The Power of Table Templates and DATA _NULL_". Proceedings of the Thirtieth Annual SAS Users Group International Conference, 30. CD-ROM. Paper 88.

SAS ODS Markup Resources: http://support.sas.com/rnd/base/topics/odsmarkup/

CONTACT INFORMATION

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

Marje Fecht
Prowerk Consulting
Marje.Fecht@prowerk.com
www.prowerk.com

Harry Droogendyk
Stratia Consulting Inc.
sesug@stratia.ca