Paper AD-226

Moving Data and Results Between SAS® and Microsoft Excel

Harry Droogendyk, Stratia Consulting Inc., Lynden, ON, Canada

ABSTRACT

Microsoft Excel spreadsheets are often the format of choice for our users, both when supplying data to our processes and as a preferred means for receiving processing results and data. SAS® offers a number of means to import Excel data quickly and efficiently. There are equally flexible methods to move data and results from SAS to Excel. This paper will outline the many techniques available and identify useful tips for moving data and results between SAS and Excel efficiently and painlessly.

INTRODUCTION

The SAS system generally provides multiple solutions to most data manipulation challenges. The task of moving data and results between SAS and Excel is no exception. This paper will provide an overview and examples of each of the many methods and will outline the pros and cons of each choice. Because SAS has such an impressive number of options available for interacting with Excel, this paper will only serve as an introduction to the available functionality, allowing further investigation by the reader to develop each method as required for their particular application.

The various methods will be considered in turn, outlining the options available for moving from SAS → Excel and vice versa. The tabular summary at the end of the paper will be a helpful resource for the reader, laying out the advantages and license requirements for each method.

TEXT DATA

Raw text data can be consumed by both SAS and Excel. Text data are often delimited, frequently by commas (CSV) or another character that is not expected to occur in the data, e.g. the tab character. Import and export steps can be written using data step code, or the SAS IMPORT and EXPORT procedures can be utilized. The ability to read and write raw text data has been a strength of Base SAS since its inception and is available to all SAS users.

```
proc export   data          = sashelp.class
              outfile       = 'c:\temp\class.txt'
              dbms          = dlm  replace;
              delimiter     = '09'x;
run;

proc import   datafile      = 'c:\temp\class.csv'
              out           = class
              dbms          = csv  replace;     getnames      = yes ;
run;
```

Additional options can be specified when IMPORTing data. By default SAS looks for column names in the first row of the data file, begins reading data in row two and looks at the first 20 rows of the input file to determine whether columns ought to be defined as character or numeric (including date or time values). The default behavior can be modified using the GETNAMES, DATAROW and GUESSINGROWS parameters. Note that while SAS can write delimited text files with an indeterminate number of rows and delimited fields, Excel will only consume rows until it reaches its version determined limit. Excel 2007 and later files (.xlsb, .xlsx, or .xlsm file name suffixes) support 16,384 columns and 1,048,576 rows in a worksheet. Pre Excel 2007 files may contain a maximum of 256 columns and 65,536 rows.

The log contents below show that PROC EXPORT actually generated a data step behind the scenes to create the tab-delimited text file.

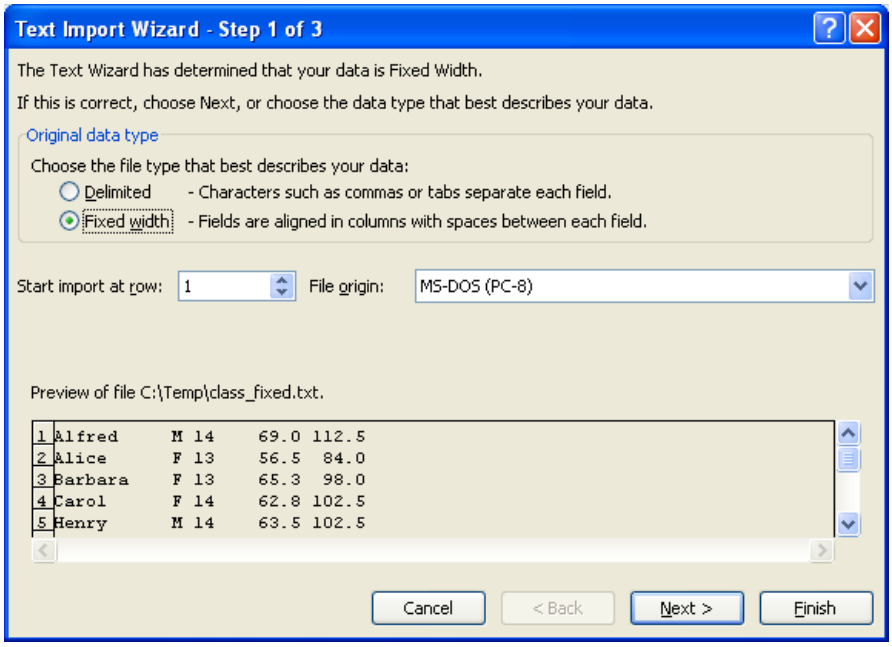
```
126      data _null_;
127      %let _EFIERR_ = 0; /* set the ERROR detection macro variable */
128      %let _EFIREC_ = 0;     /* clear export record count macro variable */
129      file 'c:\temp\class.txt' delimiter='09'x DSD DROPOVER lrecl=32767;
130      if _n_ = 1 then        / write column names or labels */
131       do;
132         put
133            "Name"
```

```
                                                  <snip>
142         ;
143       end;
144     set  SASHELP.CLASS   end=EFIEOD;
145         format Name $8. ;
                                                  <snip>
150       do;
151         EFIOUT + 1;
152         put Name $ @;
                                                  <snip>
157         ;
158       end;
159      if _ERROR_ then call symputx('_EFIERR_',1);  /* set ERROR detection macro
variable */
160      if EFIEOD then call symputx('_EFIREC_',EFIOUT);
161      run;
```

HINT **Is additional data processing required in addition to the import or export ?** As the log showed, SAS generated data step code behind the scenes to do the deed. Create most of the customized data step required by *first* coding the import or export procedure, running the procedure, and grabbing the salient data step code out of the log, adding the required data manipulation logic. Use the completed data step code in place of the SAS procedure.

Not all text data is delimited. Fixed width columnar text data can be created using data step code. The resulting file can be consumed by Excel and converted to columns using the Text Import Wizard.

```
filename fixed lrecl = 100 'c:\temp\class_fixed.txt';

data _null_;
    file fixed;
    set sashelp.class;
    put @1 name  @12 sex  @14 age 4.-L @19 height 5.1 @25 weight 5.1;
run;
```



DYNAMIC DATA EXCHANGE (DDE)

DDE was introduced in 1987 to allow one Windows program to communicate with another. While DDE technology is ancient (in terms of computer years), it is still supported by current versions of Windows and SAS. It remains a powerful tool for the adventurous SAS programmer who wishes to control Windows software, including Microsoft Excel. Using DDE, the SAS user can interact with Excel to do the following:

- read and write data directly from / into specific Excel worksheet cells or a range of cells
- execute many native Excel commands, eg.
    - create, delete or rename worksheets
    - sort, format cells
    - save workbooks
    - execute Excel macros

Excel must exist on the machine running the SAS program and the user must be able to execute system commands on that machine (XCMD configuration option) . DDE allows SAS to control Excel almost as if a user with a keyboard was interacting with the spreadsheet. While a DDE program is running, the user interface of the computer is effectively not available for use.

There are some excellent SAS / DDE resources available so the example below is simply to whet the reader's appetite. The sample code is taken directly from the SAS Online Documentation site.

```
/* This code assumes that Excel is installed in the directory specified  */
options noxwait noxsync;
x '"C:\Program Files\Microsoft Office\Office12\excel.exe"';

/* Sleep for 15 seconds to give Excel time to start.       */

data _null_;
   x=sleep(15);
run;

/* DDE link is established using MS Excel SHEET1, rows 1-20 and columns 1-3  */
filename data dde 'excel|sheet1!r1c1:r20c3';
data one;
   file data;
   do i=1 to 20;
      x=ranuni(i);
      y=x+10;
      z=x/2;
      put x y z;
   end;
run;

/* Microsoft defines the DDE topic SYSTEM to enable commands to be invoked
within Excel. */
filename cmds dde 'excel|system';

/* These PUT statements are executing Excel macro commands */
data _null_;
   file cmds;
   put '[SELECT("R1C1:R20C3")]';
   put '[SORT(1,"R1C1",1)]';
   put '[SAVE()]';
   put '[QUIT()]';
run;
```

While DDE is a finicky mechanism with poor error reporting and negligible debugging capabilities and is no longer being enhanced, it remains a powerful and flexible tool for some applications. See the Recommended Reading section for some helpful DDE links. More recent communication methods such as COM are also available but outside the scope of this paper.

SAS/ACCESS INTERFACE TO PC FILES ®

At installations where the SAS/Access product for PC file formats software is available and licensed, SAS can read, create and write Excel workbooks directly using the IMPORT and EXPORT procedures and the Excel LIBNAME engine.

If SAS is running on a 32-bit Windows system there are no functionality limitations. In installations where SAS is running on Unix, Linux or 64-bit Windows operating systems, there may be some restrictions and an additional PC Files Server may be required depending on the SAS version. See the **Supported Data Sources and Environments** documentation for 9.3 and 9.4 for further details. Users of 9.4 can specify DBMS=XLSX to access Excel files directly from Unix / Linux without having to reference the PC Files Server.

IMPORT AND EXPORT PROCEDURES

In addition to dealing with delimited text data, the IMPORT and EXPORT procedures can work directly with native Excel file formats (ie. .xls and .xls* file suffixes) subject to the requirements laid out in the *Supported Data Sources and Environments* documentation. The more generic ACCESS and DBLOAD procedures also have the ability to import and export Excel file format data.

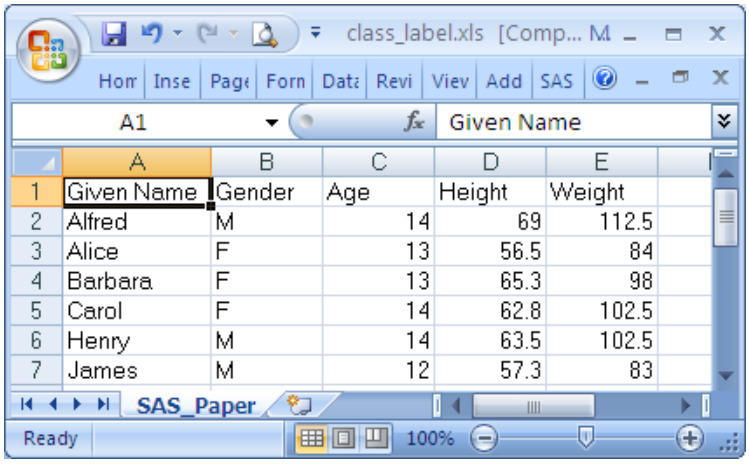
As mentioned earlier, worksheet row and column limits vary by Excel version. As one would reasonably expect, those same limits apply with using IMPORT and EXPORT with native Excel files.

PROC EXPORT

The EXPORT procedure can be used for a number of file formats (eg. MS Access, Lotus, dBase), and as a consequence has a large number of options. Only some of the EXPORT options are applicable to Excel. Further, the DBMS option specified determines which of the Excel options may be used.

```
data class;
      set sashelp.class;
      label  sex    = 'Gender'
             name   = 'Given Name';
      format height weight 9.2;
run;

proc export   data          = class
              outfile       = 'c:\temp\class_label.xls'
              dbms          = 'Excel97'   label replace;
              sheet         = 'SAS Paper'
run;
```

Note the headings for the NAME and SEX columns reflect the SAS labels specified in the data step. If the LABEL option in EXPORT was not specified, the SAS field names would have appeared in the Excel column headings. The sheet name parameter has been specified as well. Note that the SAS formats applied to the data set are NOT carried through to Excel.



	A	B	C	D	E
1	Given Name	Gender	Age	Height	Weight
2	Alfred	M	14	69	112.5
3	Alice	F	13	56.5	84
4	Barbara	F	13	65.3	98
5	Carol	F	14	62.8	102.5
6	Henry	M	14	63.5	102.5
7	James	M	12	57.3	83

Helpful PROC EXPORT options (to Excel):

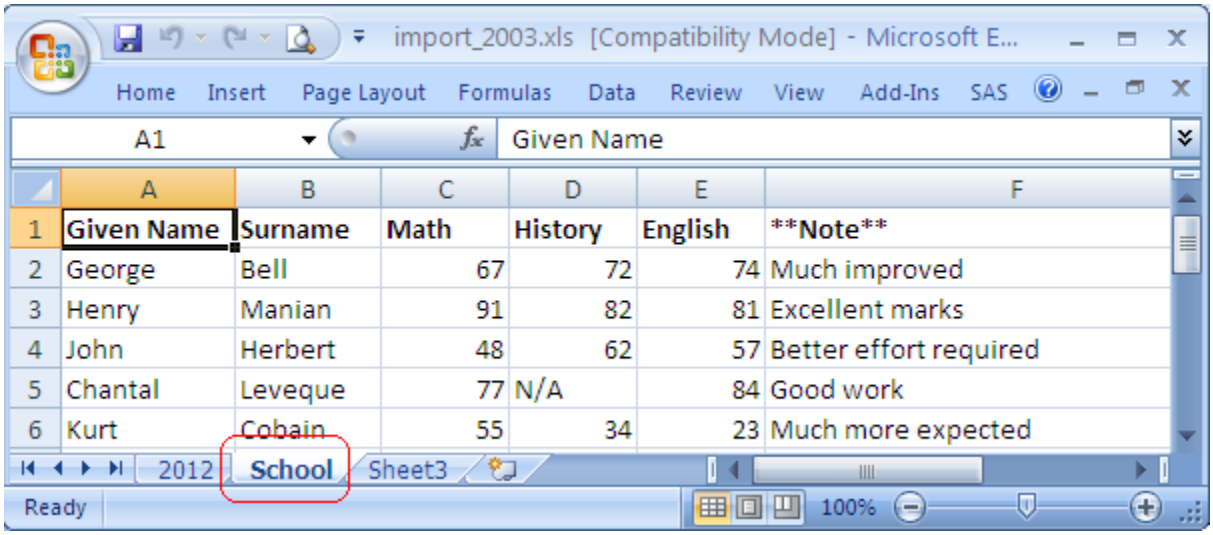
Option Name	Purpose	Values (*default*)
DBDSOPTS	Excel dataset options	DROP=
DBMS	Specify Excel version	*EXCEL*, EXCEL97, 2000, 2002, 2003, EXCEL2007, EXCEL2010, XLSX
SHEET	Names exported sheet within the workbook	*Data set name*
DATA	SAS dataset to export	
OUTFILE	Path and Excel file to create	
REPLACE	Overwrites an existing Excel file	

PROC IMPORT

IMPORT has a few more options, reflecting the flexibility often required when importing data from Excel. During the import process SAS must make decisions as to the data type and length of the Excel columns. Excel's ability to store both numeric and non-numeric data in the same column presents a challenge since SAS columns must be either character or numeric. For example, it's conceivable that an entry of 'N/A' could be entered in place of a numeric value in an Excel column where the entry was not applicable to the row, e.g. Age Licensed for a child. The use of options like GUESSINGROWS and MIXED will be helpful in circumstances where columns contain mixed data.

The somewhat bewildering array of IMPORT options and their (in)ability to work together as one might expect can make it necessary to post-process the resulting dataset after importing Excel data where columns contain mixed data types.

Importing From Excel 2003 files



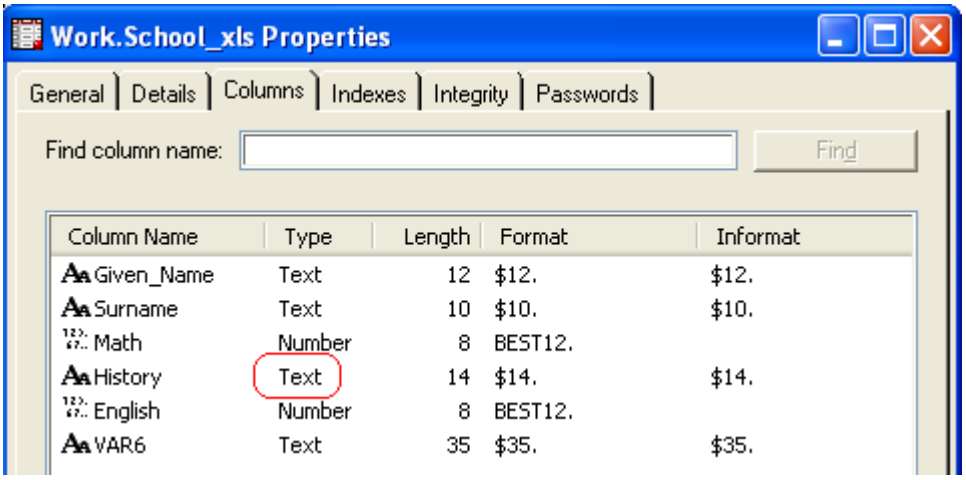

```
proc import   datafile      = 'c:\temp\import_2003.xls'
              out           = school_xls
              dbms          = 'xls'              replace ;
              sheet         = 'School';
              getnames      = yes;
              guessingrows  = 20;
run;
```

For certain DBMS option values, the SAS log informs where column names have been altered to conform them to standard SAS column names:

```
NOTE:    Variable Name Change.  Given Name -> Given_Name
```

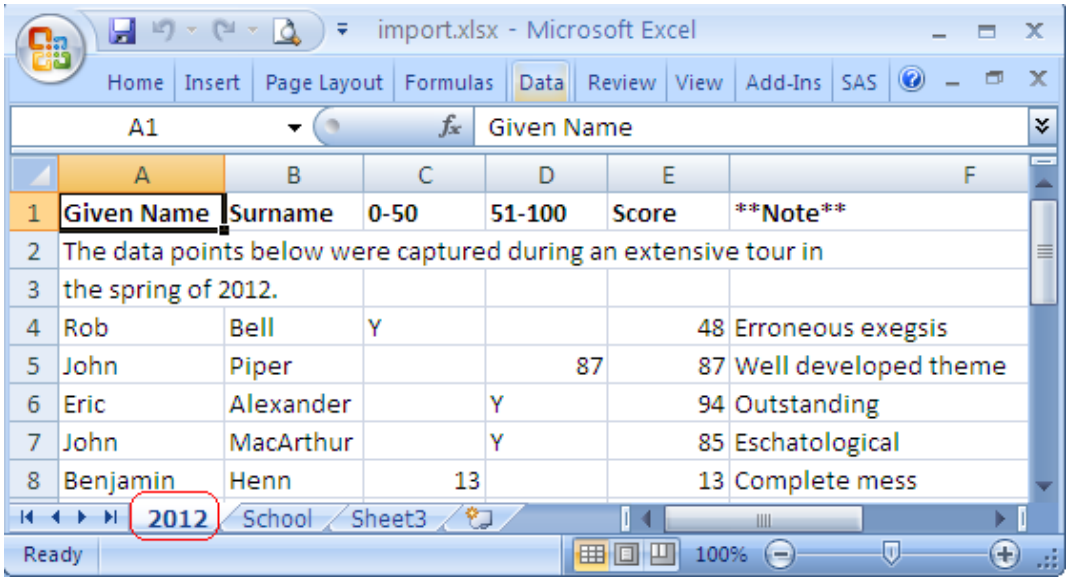
```
NOTE:    Variable Name Change.  **Note** -> VAR6
NOTE: The import data set has 5 observations and 6 variables.
NOTE: WORK.SCHOOL_XLS data set was successfully created.
```

Note that History is a character variable because the “N/A” value was found within the first number of rows (i.e. GUESSINGROWS value) in an otherwise numeric column. If the first number of rows had all been numeric and the “N/A” was found in a row beyond the GUESSINGROWS value, the column would have been defined as numeric and the History value would have been set to missing for the observation with “N/A”.



Column Name	Type	Length	Format	Informat
Given_Name	Text	12	$12.	$12.
Surname	Text	10	$10.	$10.
Math	Number	8	BEST12.	
History	Text	14	$14.	$14.
English	Number	8	BEST12.	
VAR6	Text	35	$35.	$35.

Importing from Excel 2007 (and later) files



	A	B	C	D	E	F
1	Given Name	Surname	0-50	51-100	Score	**Note**
2	The data points below were captured during an extensive tour in					
3	the spring of 2012.					
4	Rob	Bell	Y		48	Erroneous exegsis
5	John	Piper		87	87	Well developed theme
6	Eric	Alexander		Y	94	Outstanding
7	John	MacArthur		Y	85	Eschatological
8	Benjamin	Henn	13		13	Complete mess

When importing data from this spreadsheet, rows 2 and 3 must be skipped since they do not contain the desired data. Using the DBDSOPTS option, the FIRSTOBS value can be specified to define the location of the first data row. This option is only available when importing from Excel 2007/2010 files and the DBMS value specifies EXCEL2007 or EXCEL2010. Unfortunately the DATAROW option used when importing delimited text data is not available for Excel files.

```
proc import   datafile     = 'c:\temp\import.xlsx'
              out          = import
              dbms         = excel2007          replace ;
              sheet        = '2012';            getnames      = yes;
              DBDSOPTS     = 'firstobs=3';
```

```
run;
```

Not only do the different DBMS values determine available options and procedure functionality, the log messages differ as well for each. In this case, the column name alterations are not mentioned, nor is the number of imported observations reported in the log.

```
NOTE: WORK.IMPORT data set was successfully created.
```

DBDSOPTS has been specified to skip over rows 2 and 3. Note that the first row containing the column names is not counted when specifying FIRSTOBS. When SAS alters the non-standard column names, the results are sometimes different than one might expect. Where the first character is non-standard, it is replaced with an underscore. In this case, the names of the columns no longer reflect what they contain as defined by the Excel file. All columns have been defined as character data except Score which contained only numeric data. Again, post-import processing is likely required to "clean up" this dataset.

VIEWTABLE: Work.Import

	Given_Name	Surname	__50	_1_100	Score	__Note__
1	Rob	Bell	Y		48	Erroneous exegsis
2	John	Piper		87	87	Well developed theme
3	Eric	Alexander		Y	94	Outstanding
4	John	MacArthur		Y	85	Eschatological
5	Benjamin	Henn	13		13	Complete mess

Helpful Excel options for PROC IMPORT. Please see the SAS 9.4 documentation for a fuller description of the various options available:

Option Name	Purpose	Values (*default*)
VERSION	Defines version of Excel cell, also determines how other options perform (or not)	*97 for .xls files*, 2010, 2007, 2003, 2002, 2000, 97, 95, and 5
RANGE	Specific range within the workbook	
SHEET	Specific sheet within the workbook	
DBDSOPTS	Excel dataset options - not available when DBMS=XLS	FIRSTOBS=n (useful when data does not start on row 2) OBS=nn (must be GE firstobs) DROP=
GETNAMES	Retrieve column names from first row	*YES* | NO
MIXED	Convert all numeric values to character where a column contains both numeric and character (may not be used with DBMS='XLSX')	YES | *NO*
SCANTEXT	Scan the column to determine character field length	*YES* | NO
SCANTIME	Assign TIME. format where applicable	*YES* | NO
USEDATE	Assigns date9. format to Excel date columns	*YES* | NO
GUESSINGROWS	Number of rows to scan to determine column type - may be specified with DBMS='XLS' - illegal for DBMS='XLSX'	Windows registry contains GUESSINGROWS setting as well

EXCEL LIBNAME ENGINE

Since SAS version 9, the EXCEL libname engine has been available to installations that license the SAS Access to PC Files software. This engine allows programs to interact with Excel workbooks almost as if they are native SAS datasets. Excel libnames can be used with virtually any SAS procedure and with the data step.

The Excel libname syntax is quite familiar:

```
libname xls <engine-name> physical-file-name;  eg.
libname xls excel 'c:\temp\report.xls';
libname xls xlsx '/users/droog/report.xlsx';
    … SAS code …
libname xls clear;      * very important !! ;
```

If the physical file specified does not exist, SAS will create a workbook of that name. Once the libname statement is successfully executed, the Excel workbook and the sheets and named ranges defined in the workbook will be available in the SAS Explorer window.

The library can be treated like any other SAS libref with a couple of caveats:

- row and column limitations as defined by Excel, eg. limitations on number of rows and columns depending on Excel version
- Excel sheet and column names can contain characters not normally considered valid by SAS, use of options validvarname=any may be necessary when reading from Excel using the libname.

To reference Excel sheet names and named ranges, the familiar libref.dataset syntax is used (sheet names are suffixed with $, necessitating the use of SAS Name Literals syntax, ie. 'sheet1$'n when referencing). eg:

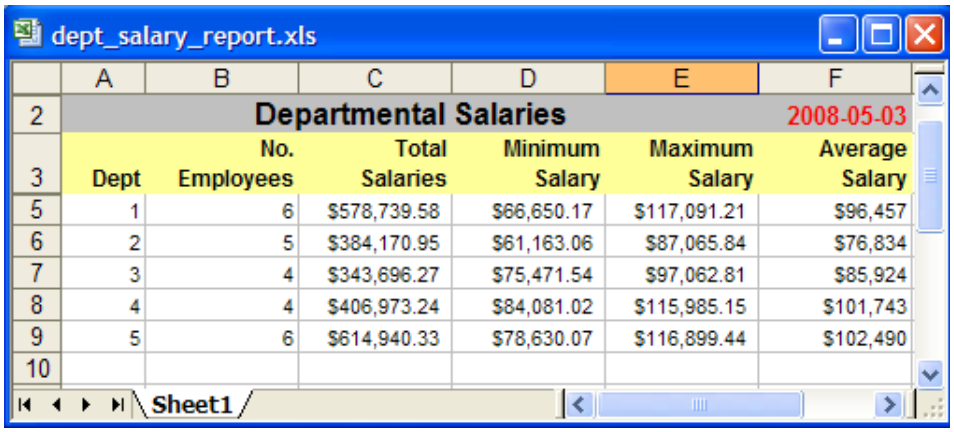
```
data xls.'sheet1$'n;       * referencing worksheet sheet1 ;
data xls.range_out;        * referencing named range range_out;
```

Writing SAS data to Excel

To populate Excel spreadsheets with the libname engine, almost any DATA or PROC step can be used, eg.

```
data xls.class;
   set sashelp.class;
run;
proc copy in     = sashelp
          out    = xls;
   select prdsale shoes;
run;
```

The use of Excel "named ranges" and the Excel libname engine, provide flexible output options that yield publishable reports (see example in screenshot immediately below). For a fuller treatment of this subject matter, please see http://www.stratia.ca/papers/excel_libname.pdf.



dept_salary_report.xls

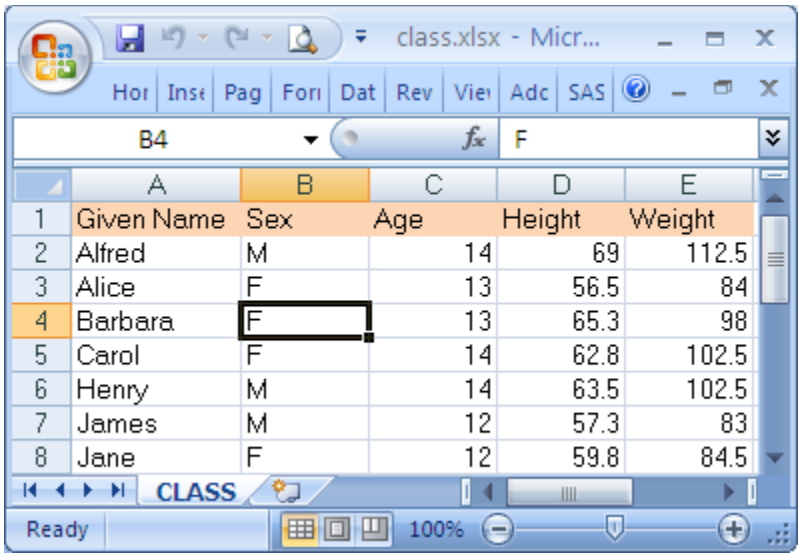
	A	B	C	D	E	F
2	Departmental Salaries					2008-05-03
3	Dept	No. Employees	Total Salaries	Minimum Salary	Maximum Salary	Average Salary
5	1	6	$578,739.58	$66,650.17	$117,091.21	$96,457
6	2	5	$384,170.95	$61,163.06	$87,065.84	$76,834
7	3	4	$343,696.27	$75,471.54	$97,062.81	$85,924
8	4	4	$406,973.24	$84,081.02	$115,985.15	$101,743
9	5	6	$614,940.33	$78,630.07	$116,899.44	$102,490
10						

Sheet1

Reading Excel data into SAS

In similar fashion, the Excel libname engine can be utilized by SAS data and procedure steps to read tabular Excel data. The code below is reading from the "Class" worksheet in the class.xlsx workbook.

```
libname xls excel 'c:\temp\class.xlsx';

data my_class;
      set xls.'class$'n;
run;

proc print data = my_class label noobs ;
run;

libname xls clear;
```



class.xlsx - Micr...

B4 F

	A	B	C	D	E
1	Given Name	Sex	Age	Height	Weight
2	Alfred	M	14	69	112.5
3	Alice	F	13	56.5	84
4	Barbara	F	13	65.3	98
5	Carol	F	14	62.8	102.5
6	Henry	M	14	63.5	102.5
7	James	M	12	57.3	83
8	Jane	F	12	59.8	84.5

CLASS

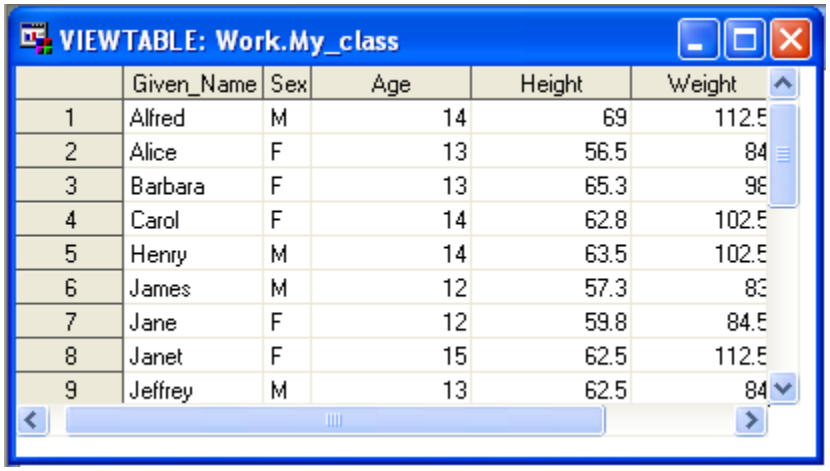
Ready 100%

Log:

```
2    libname xls excel 'c:\temp\class.xlsx';

NOTE: Libref XLS was successfully assigned as follows:
      Engine:        EXCEL
      Physical Name: c:\temp\class.xlsx
3    data my_class;
```

```
4        set xls.'class$'n;
5    run;

NOTE: There were 19 observations read from the data set XLS.'class$'n.
NOTE: The data set WORK.MY_CLASS has 19 observations and 5 variables.
```

The “Given Name” Excel column heading has been converted into a standard SAS column name, but as the output results show, the original Excel column name has been preserved as a SAS column label.



	Given_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

Output:

```
Given
Name        Sex     Age     Height     Weight

Alfred       M       14      69.0       112.5
Alice        F       13      56.5        84.0
```

Limitations of the Excel Libname:

While the Excel libname is more powerful than PROC EXPORT / IMPORT, it does not have the full range of functionality provided by DDE.

Excel Libname Capabilities	Not Possible Via Excel Libname
• creating new workbooks • creating new sheets and named ranges • deleting existing data within a named range • populating an existing, empty named range • appending data to an existing named range • reading data from an existing sheet or named range	• formatting changes, eg. font, color • deleting an entire workbook, sheets or named ranges • deleting cells containing a formula • writing a formula into a cell

Excel Libname options:

Many of the same options already seen in the PROC IMPORT / EXPORT sections surface again for the Excel libname engine, while others have been renamed. The following options are the most popular, see the SAS Online documentation for a complete list of options available.

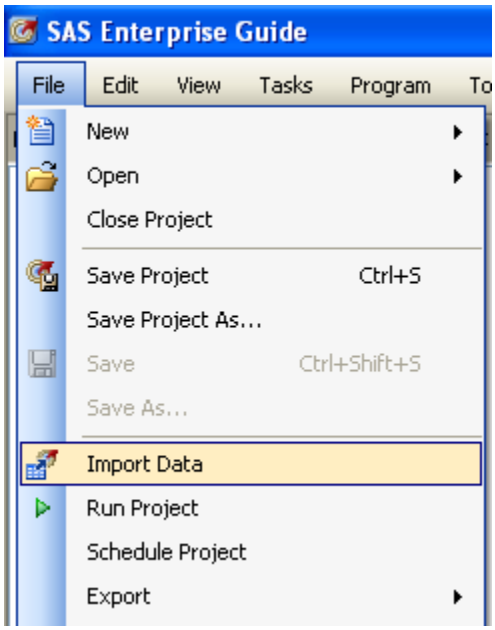
Option Name	Purpose	Values (*default*)
DBLABEL	Use SAS column labels to create Excel column headers rather than the column name	YES | *NO*

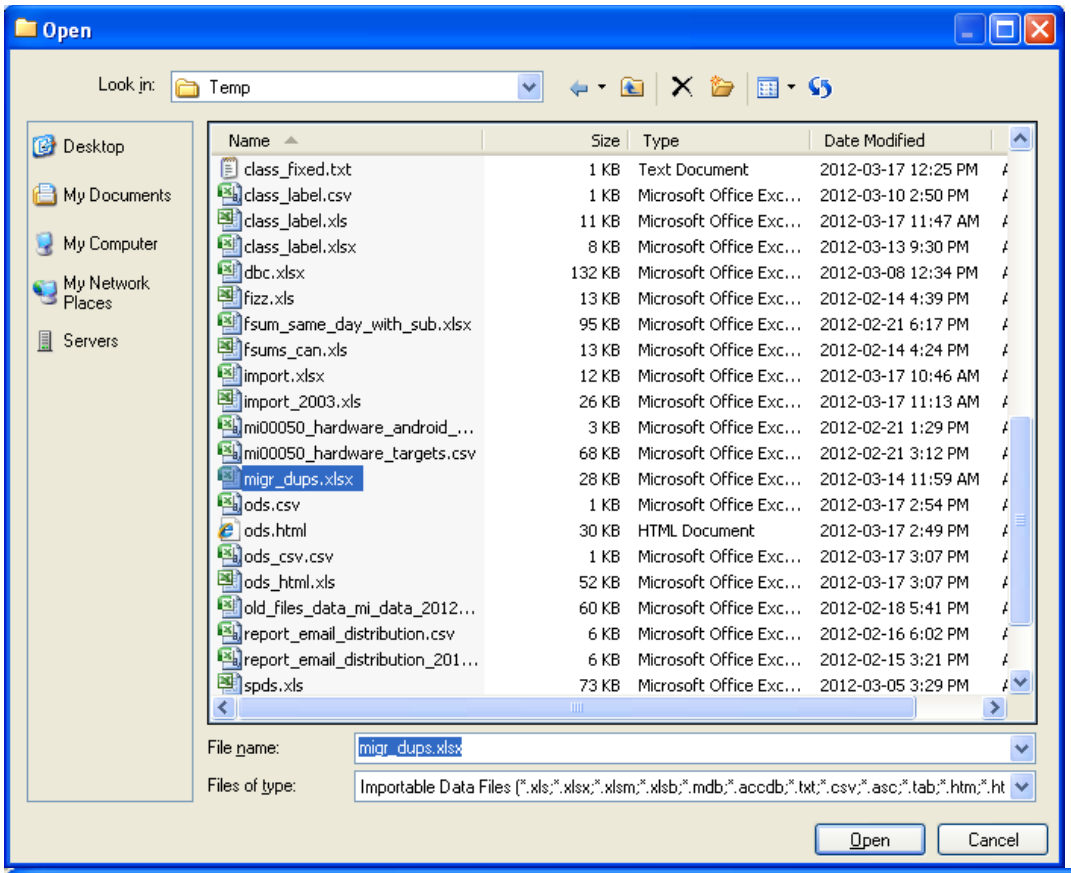
HEADER	Use Excel column headers to name SAS columns	*YES* | NO
MIXED	Convert numeric values to character for mixed type columns	YES | *NO*
DBSASTYPE	Define type and length for specific columns	e.g. DBSASTYPE=(VarName='CHAR(8)')
VERSION	Specify Excel version	Only necessary when creating Excel files
SCAN_TEXT	Must be set to NO to append to an existing Excel workbook	YES | *NO*

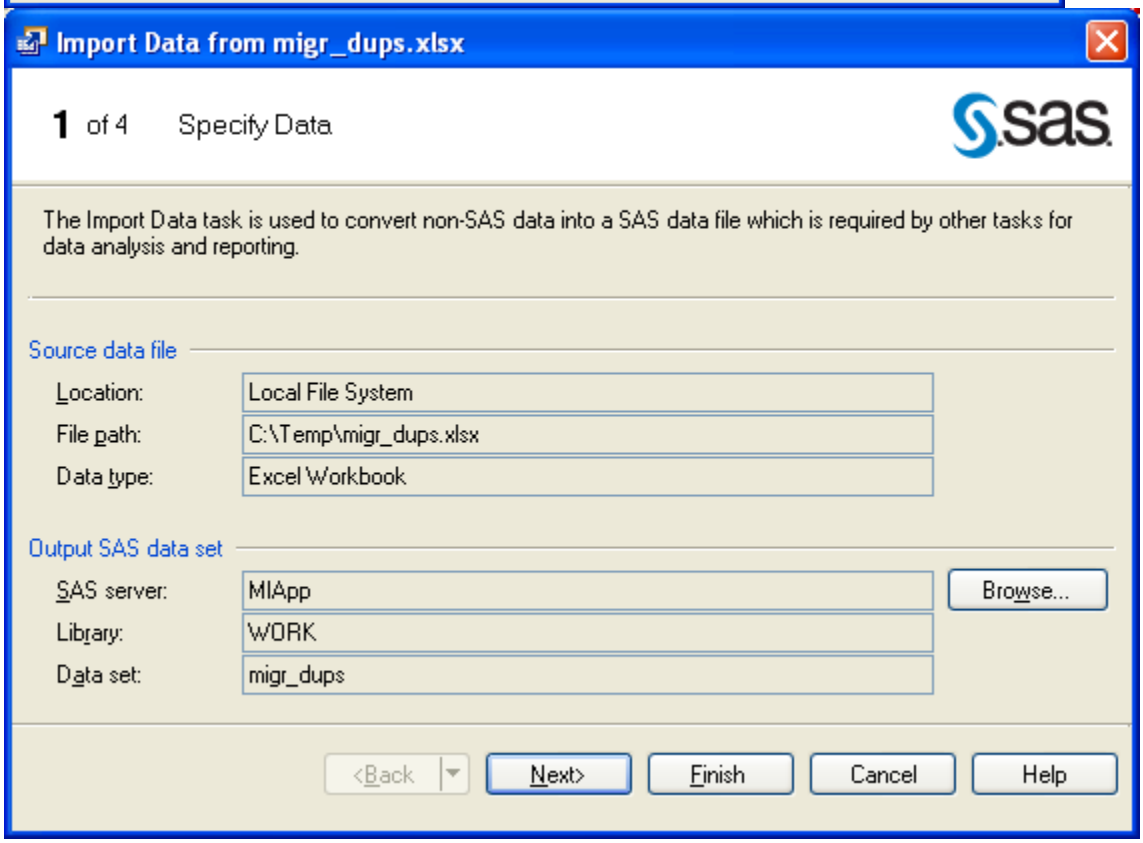
SAS ENTERPRISE GUIDE ®

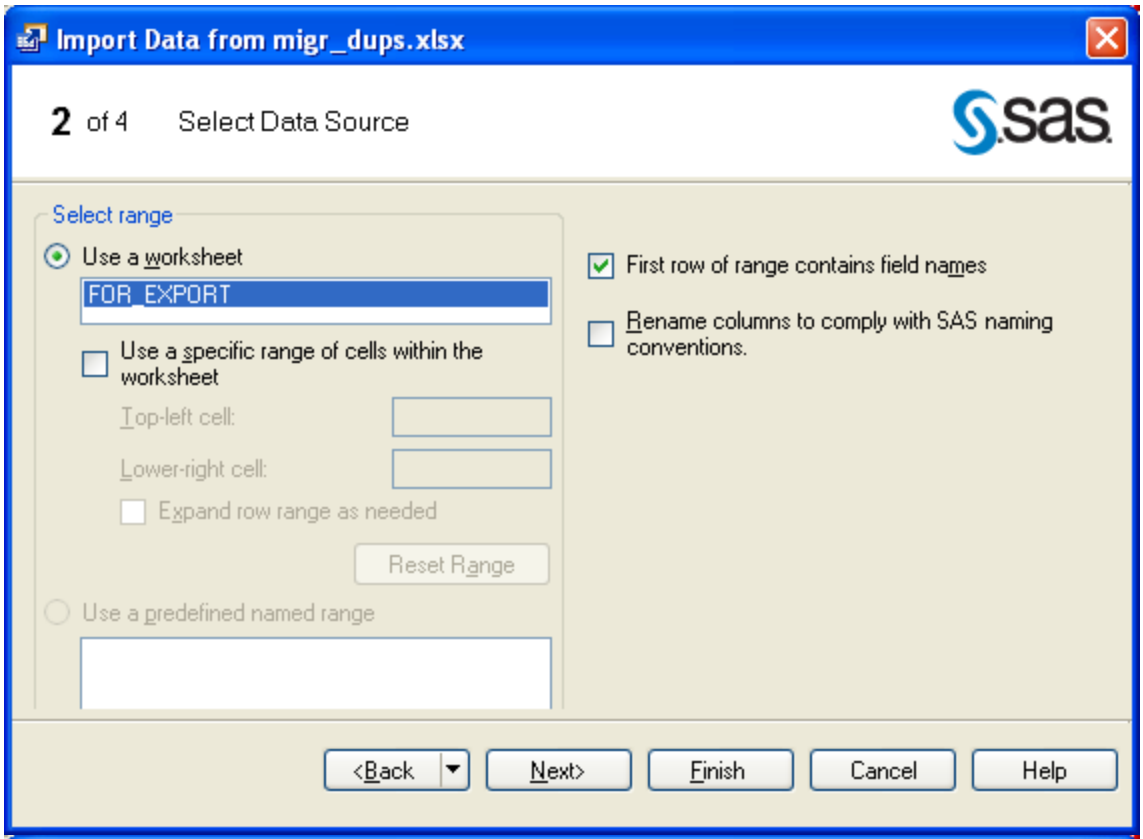
SAS Enterprise Guide (EG) has built-in functionality that allows users to import and export Excel data even if SAS Access for PC Files is not licensed. Both wizards are available from the File menu bar choice.

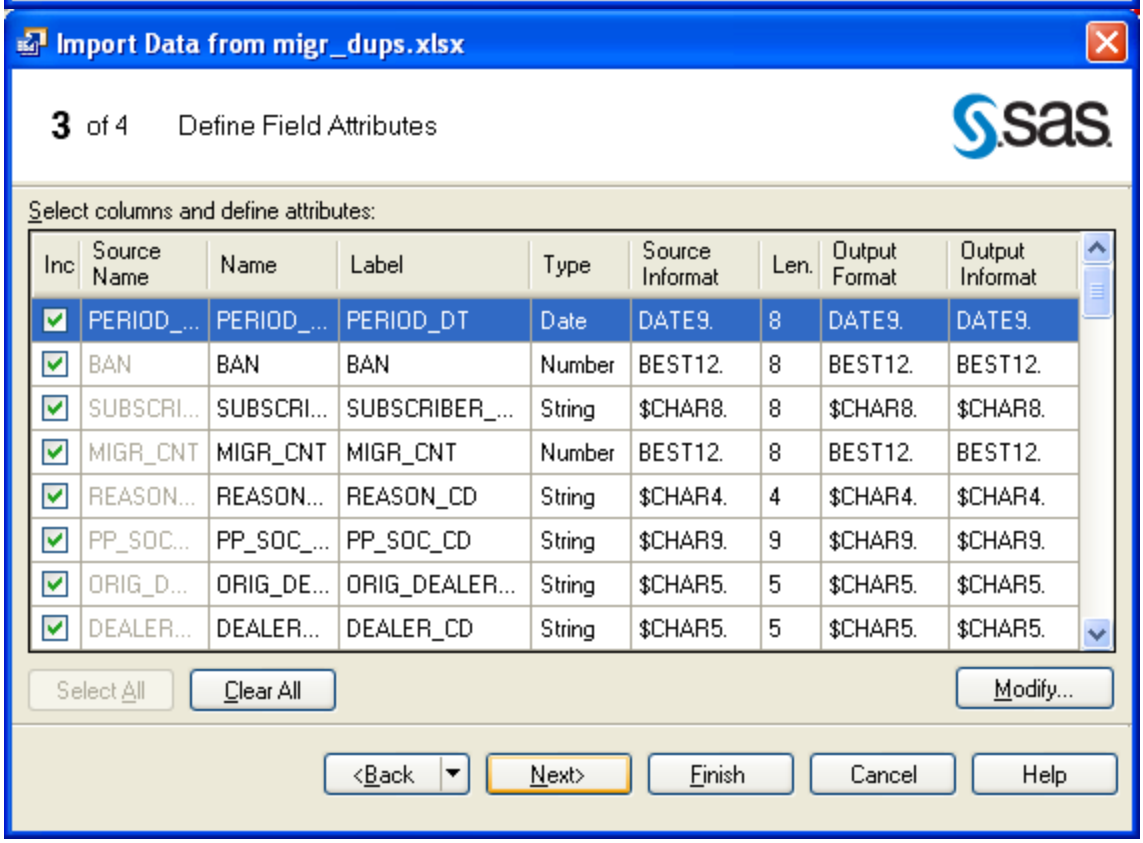
IMPORTING DATA USING EG

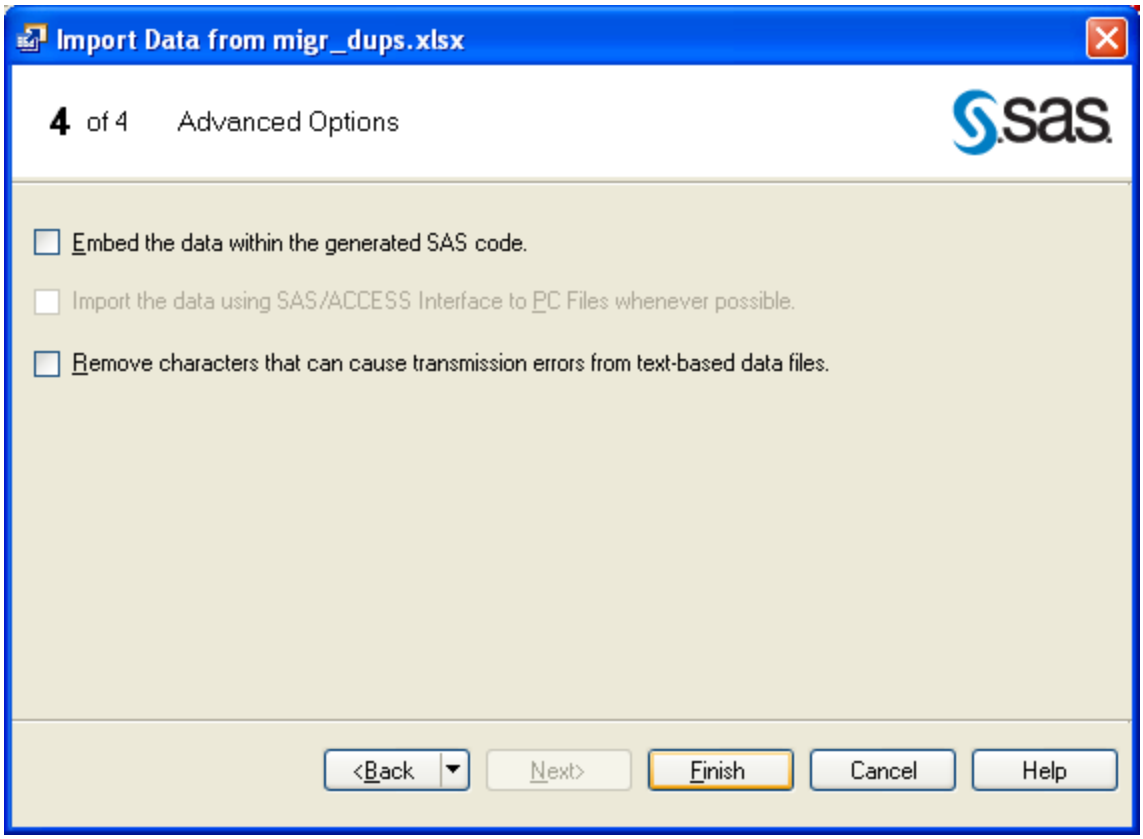



Open
Look in: Temp
Desktop
My Documents
My Computer
My Network Places
Servers
Name Size Type Date Modified
class_fixed.txt 1 KB Text Document 2012-03-17 12:25 PM
class_label.csv 1 KB Microsoft Office Exc... 2012-03-10 2:50 PM
class_label.xls 11 KB Microsoft Office Exc... 2012-03-17 11:47 AM
class_label.xlsx 8 KB Microsoft Office Exc... 2012-03-13 9:30 PM
dbc.xlsx 132 KB Microsoft Office Exc... 2012-03-08 12:34 PM
fizz.xls 13 KB Microsoft Office Exc... 2012-02-14 4:39 PM
fsum_same_day_with_sub.xlsx 95 KB Microsoft Office Exc... 2012-02-21 6:17 PM
fsums_can.xls 13 KB Microsoft Office Exc... 2012-02-14 4:24 PM
import.xlsx 12 KB Microsoft Office Exc... 2012-03-17 10:46 AM
import_2003.xls 26 KB Microsoft Office Exc... 2012-03-17 11:13 AM
mi00050_hardware_android_... 3 KB Microsoft Office Exc... 2012-02-21 1:29 PM
mi00050_hardware_targets.csv 68 KB Microsoft Office Exc... 2012-02-21 3:12 PM
migr_dups.xlsx 28 KB Microsoft Office Exc... 2012-03-14 11:59 AM
ods.csv 1 KB Microsoft Office Exc... 2012-03-17 2:54 PM
ods.html 30 KB HTML Document 2012-03-17 2:49 PM
ods_csv.csv 1 KB Microsoft Office Exc... 2012-03-17 3:07 PM
ods_html.xls 52 KB Microsoft Office Exc... 2012-03-17 3:07 PM
old_files_data_mi_data_2012... 60 KB Microsoft Office Exc... 2012-02-18 5:41 PM
report_email_distribution.csv 6 KB Microsoft Office Exc... 2012-02-16 6:02 PM
report_email_distribution_201... 6 KB Microsoft Office Exc... 2012-02-15 3:21 PM
spds.xls 73 KB Microsoft Office Exc... 2012-03-05 3:29 PM
File name: migr_dups.xlsx
Files of type: Importable Data Files (*.xls;*.xlsx;*.xlsm;*.xlsb;*.mdb;*.accdb;*.txt;*.csv;*.asc;*.tab;*.htm;*.ht
Open
Cancel

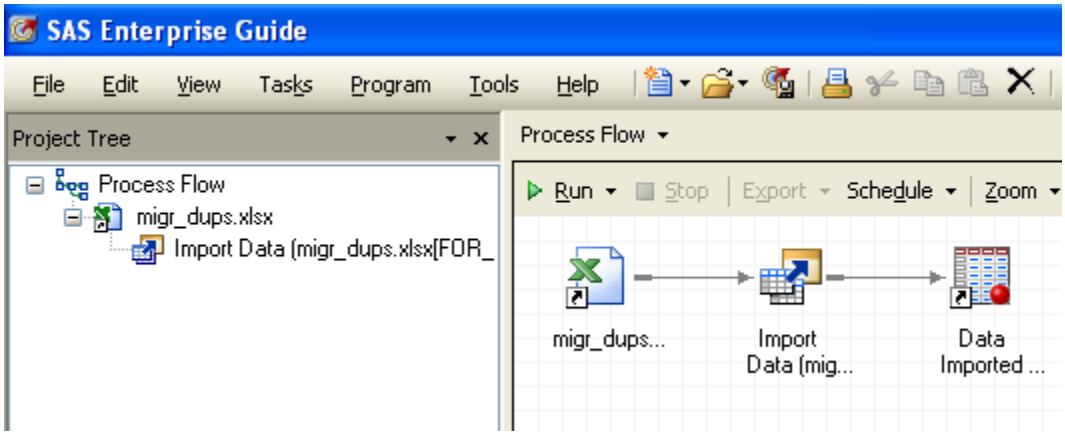

Import Data from migr_dups.xlsx
1 of 4 Specify Data
sas
The Import Data task is used to convert non-SAS data into a SAS data file which is required by other tasks for data analysis and reporting.
Source data file
Location: Local File System
File path: C:\Temp\migr_dups.xlsx
Data type: Excel Workbook
Output SAS data set
SAS server: MIApp
Library: WORK
Data set: migr_dups
Browse...
<Back
Next>
Finish
Cancel
Help


Import Data from migr_dups.xlsx
2 of 4 Select Data Source
Select range
Use a worksheet
FOR_EXPORT
Use a specific range of cells within the worksheet
Top-left cell:
Lower-right cell:
Expand row range as needed
Reset Range
Use a predefined named range
First row of range contains field names
Rename columns to comply with SAS naming conventions.
<Back
Next>
Finish
Cancel
Help


Import Data from migr_dups.xlsx
3 of 4 Define Field Attributes
Select columns and define attributes:
Inc
Source Name
Name
Label
Type
Source Informat
Len.
Output Format
Output Informat
PERIOD_... PERIOD_... PERIOD_DT Date DATE9. 8 DATE9. DATE9.
BAN BAN BAN Number BEST12. 8 BEST12. BEST12.
SUBSCRI... SUBSCRI... SUBSCRIBER_... String $CHAR8. 8 $CHAR8. $CHAR8.
MIGR_CNT MIGR_CNT MIGR_CNT Number BEST12. 8 BEST12. BEST12.
REASON... REASON... REASON_CD String $CHAR4. 4 $CHAR4. $CHAR4.
PP_SOC... PP_SOC_... PP_SOC_CD String $CHAR9. 9 $CHAR9. $CHAR9.
ORIG_D... ORIG_DE... ORIG_DEALER... String $CHAR5. 5 $CHAR5. $CHAR5.
DEALER... DEALER... DEALER_CD String $CHAR5. 5 $CHAR5. $CHAR5.
Select All
Clear All
Modify...
<Back
Next>
Finish
Cancel
Help



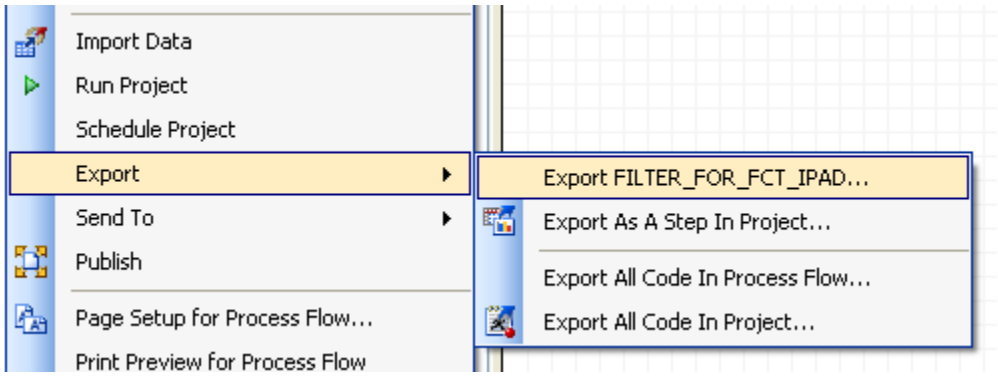
EG Project with Excel Import Step



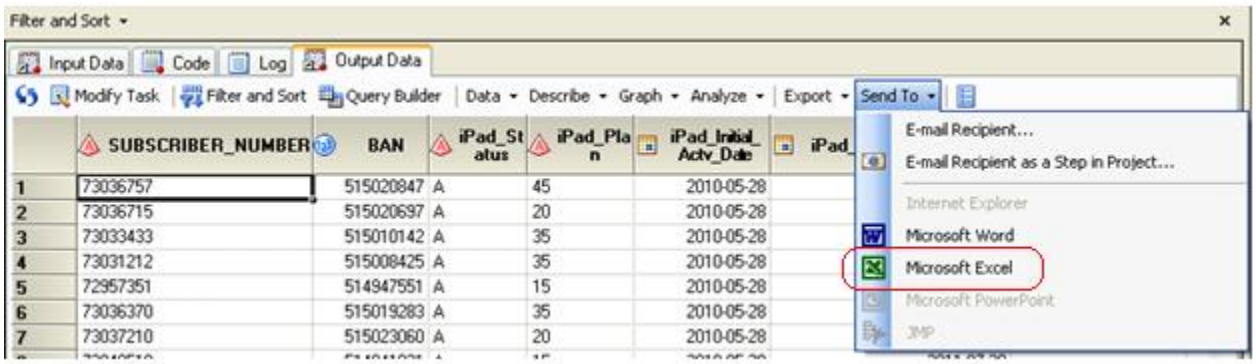
EXPORTING DATA USING EG

EG has several methods to export data to Excel. And, since EG is able to access data tables from many different back-end DB systems where the appropriate SAS Access library has been defined, any data from any library accessible by EG can be exported to Excel.

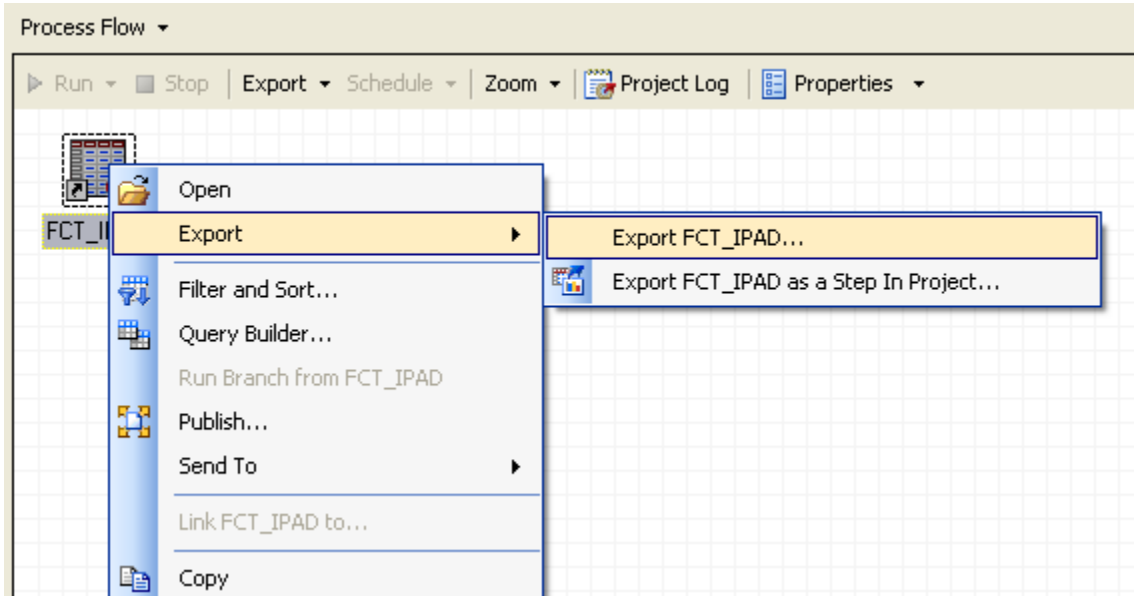
Export from the File Menu:

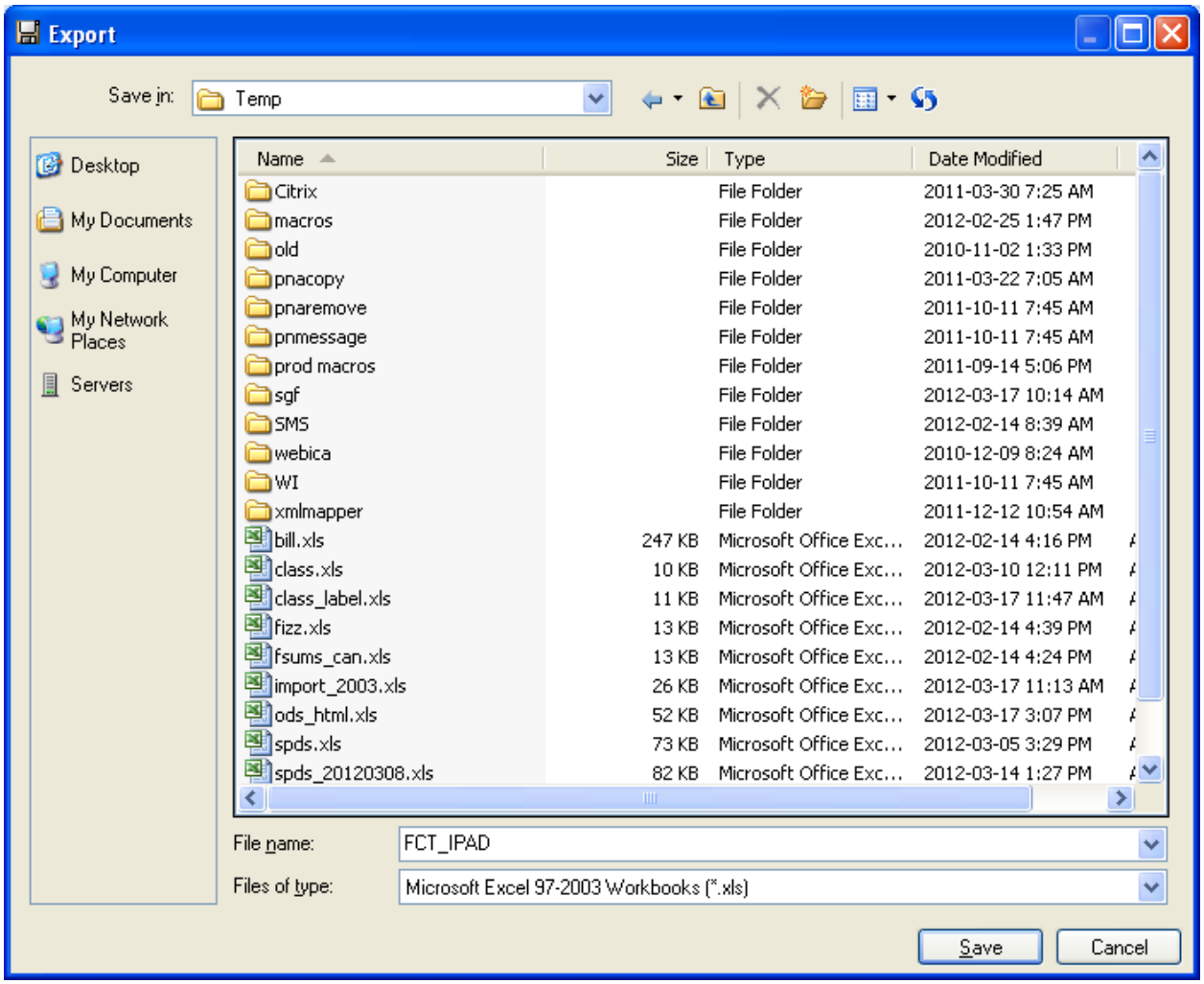


Export while viewing data:



Export by right-clicking on the data on the project palette:





OUTPUT DELIVERY SYSTEM (ODS)

The SAS Output Delivery System has a number of output destinations that create files consumable by Excel, notably: CSV, HTML, and at least two Tagsets: ExcelXP, MSOffice2K. In v9.4M1 the ODS EXCEL destination was introduced to create files in native Microsoft Excel 2010 XLXS format. ODS EXCEL also supports Excel graphics and both A1 and R1C1 formula notations.

The use of ODS styles (especially when customized using PROC TEMPLATE) allow the tailoring of the output to suit the users' purposes. The tagsets provide a number of options for seamlessly creating publishable output. In addition, field attributes can be defined for specific Excel formatting, traffic-lighting of results and calculated cells by including Excel formulas in the output.

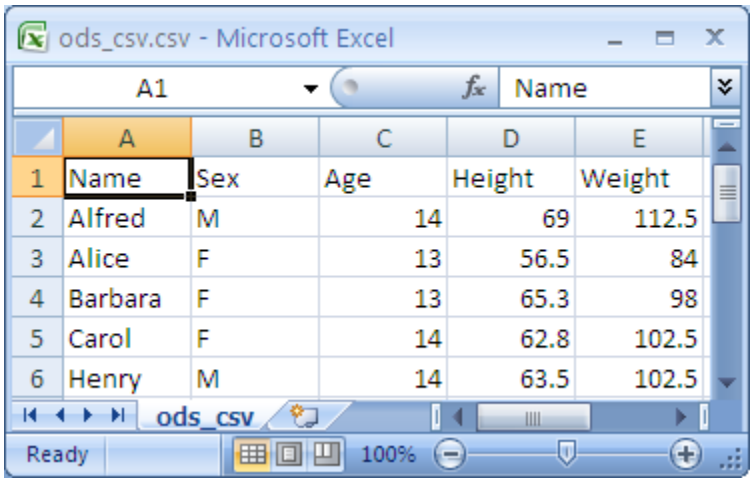
While the nuances of each particular ODS destination are outside the scope of this paper, links in the recommended reading section point to additional useful information.

CSV files are opened by Excel as per the Windows default behavior. HTML files with a .xls* suffix are very happily consumed by Excel and displayed as Excel workbooks. The XML created by the tagsets aligns with Microsoft's presentation standards for Excel 2002 and beyond. The native Excel format created by the EXCEL destination open without the warning generated for the tagset produced files ("The file you are trying to open, 'blah.xls', is in a different format than specified by the file extension..") and can be imported by SAS using PROC IMPORT, i.e. ODS EXCEL creates real Excel files – FINALLY!!! ☺

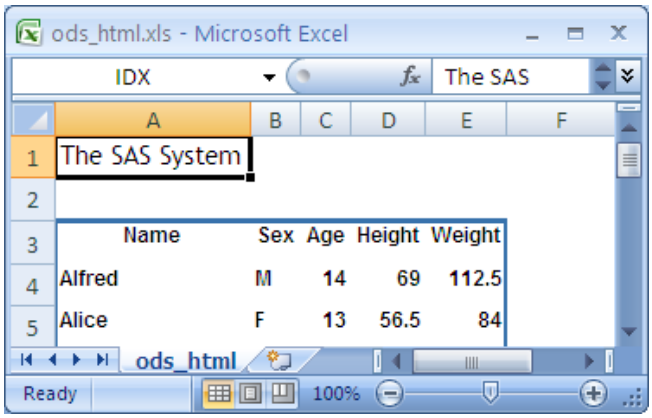
SAS ODS code to create Excel consumable output:

```
ods results=off;    ods listing close;

ods csv file  = '$HOME/ods_csv.csv';
ods html file = '$HOME/ods_html.xls' style=seaside;
ods tagsets.excelxp file = '$HOME/tagsets_excelxp.xls'
      options ( autofilter='all' ) style=ocean;
ods excel file = '$HOME/ods_excel.xlsx'
      options ( sheet_name='real Excel' ) style=seaside;

proc print data = sashelp.class noobs label;
run;

ods csv close;            ods html close;
ods tagsets.excelxp close;
ods excel close;

ods listing;        ods results=on;
```

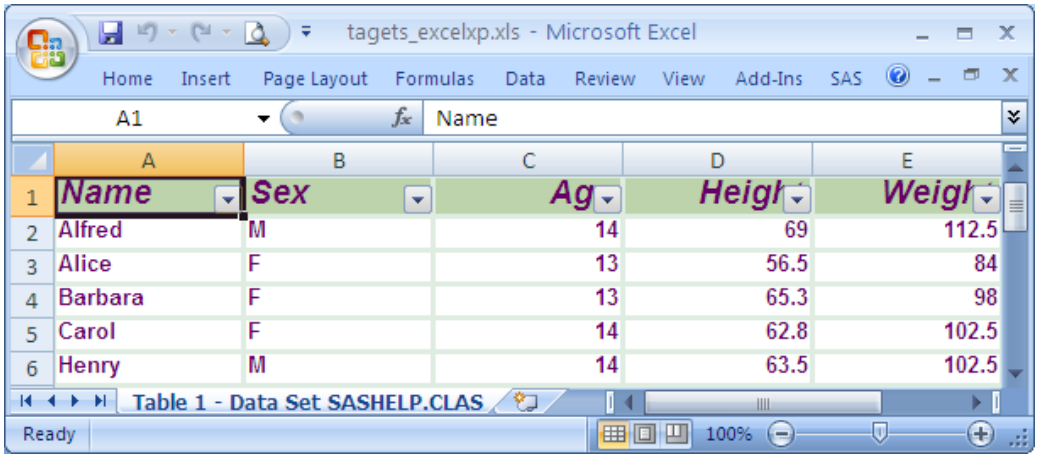
ODS CSV



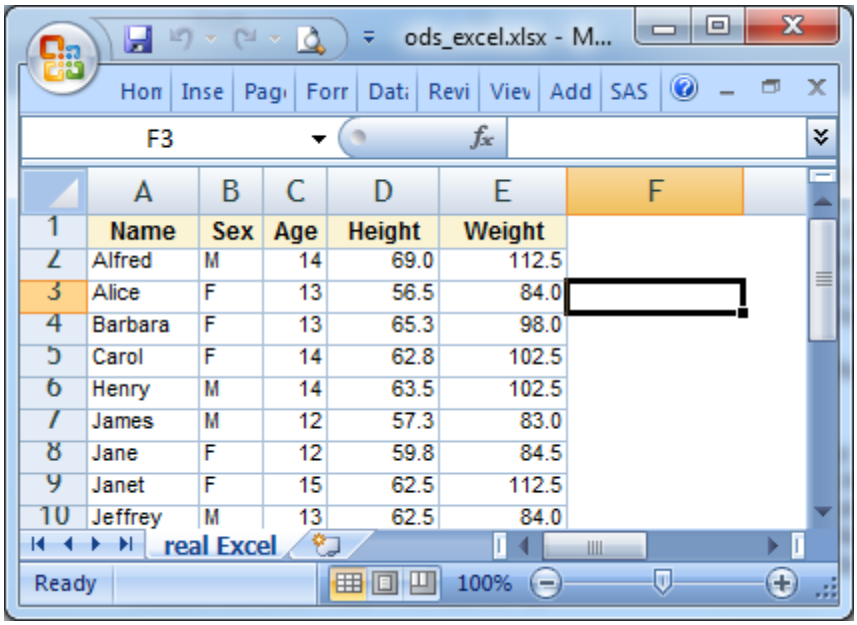
ODS HTML



ODS tagsets.ExcelXP (note the auto-filters):



ODS EXCEL (note the sheet name):



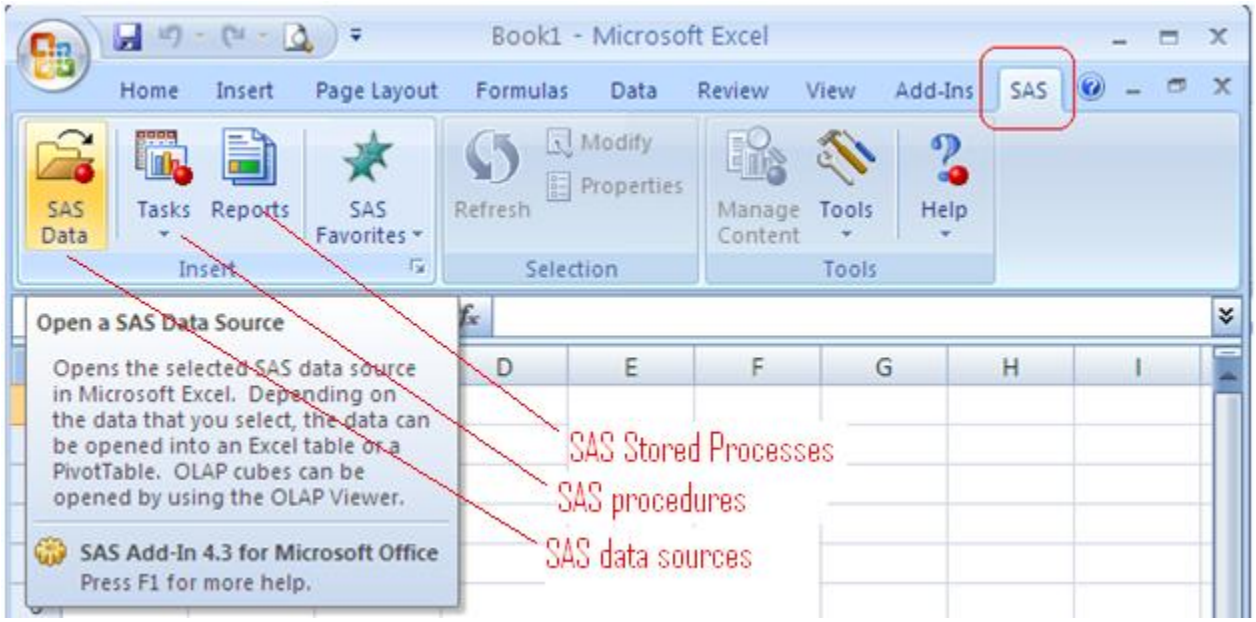
ADD-IN FOR MICROSOFT OFFICE ®

The Add-in for Microsoft Office (AMO), available as part of the SAS Business Intelligence Suite ®, also provides a mechanism to pull data from SAS resources into Excel. The metadata undergirding the SAS BI platform allows almost any RDBMS, SAS dataset or text file to be defined and accessed as if it were a simple SAS dataset. AMO, tightly coupled to the Microsoft product, allows easy interaction between Excel and the metadata-defined data store. In addition, SAS resources available to Excel from local or network drives may be manipulated using AMO.

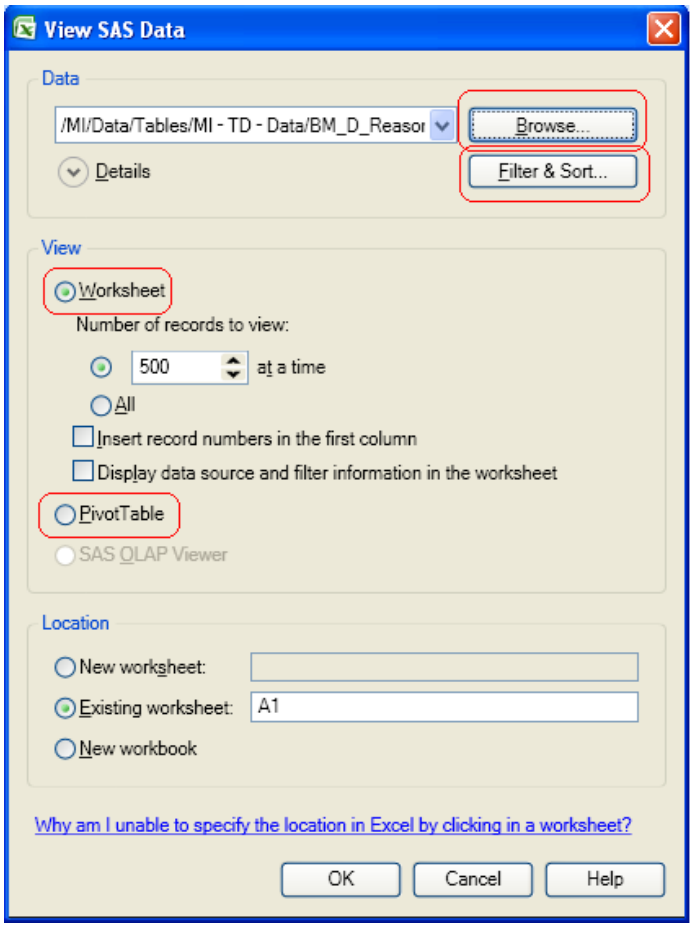
If AMO is available, “SAS” will be one of the menu bar choices in Excel. In Excel 2007 or later, the SAS ribbon is also available.

AMO has a number of features that not only allow the consumption of data in detail form, but can also import directly into Excel Pivot Tables. In addition, SAS Stored Processes can be invoked via the Reports button to populate the spreadsheet with runtime results. A large number of SAS tasks familiar to SAS Enterprise Guide ® users is accessible via the Tasks button to run SAS procedures directly from Excel on enterprise data.

Finally… Excel can do analytics !!! ☺

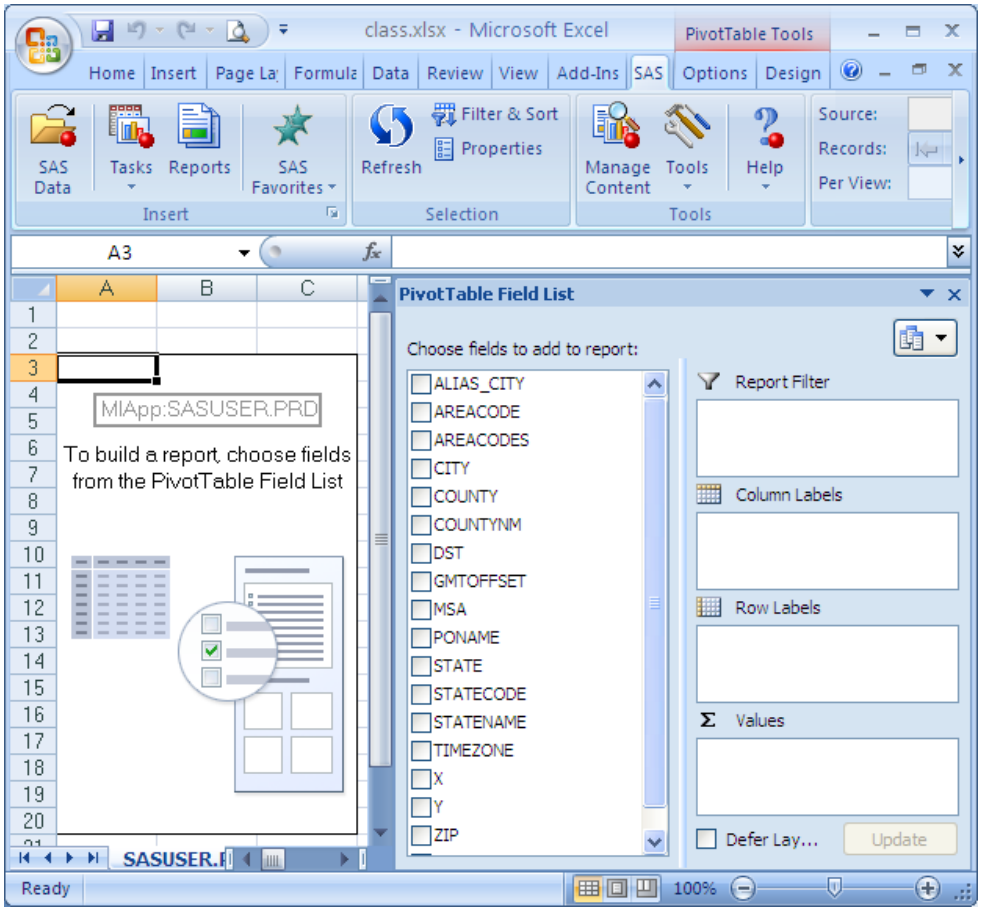


To browse data from Excel click the SAS Data button.

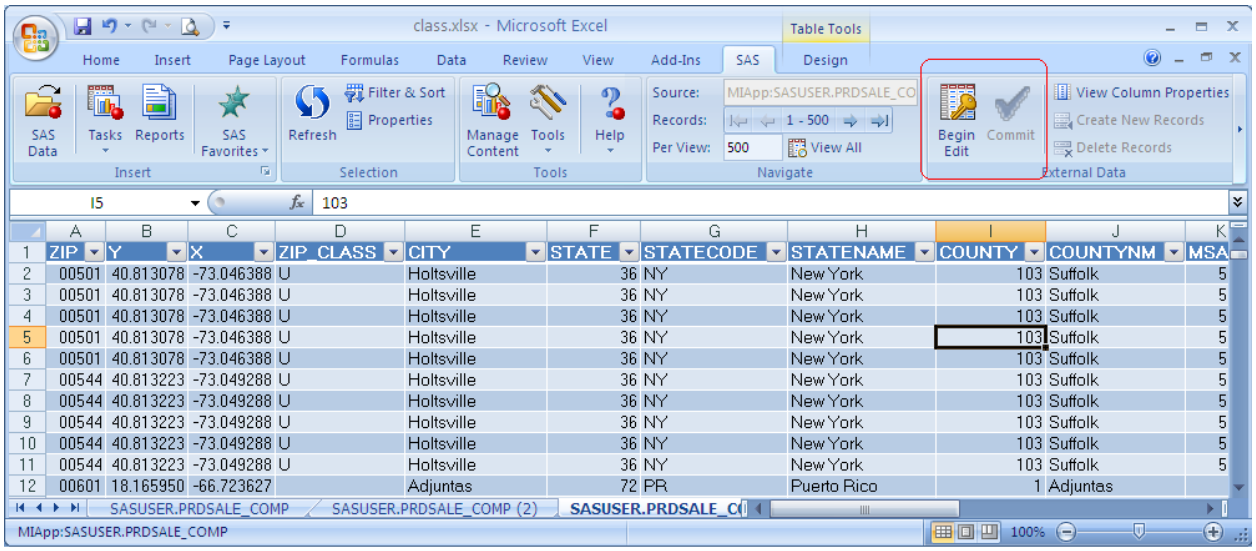


Click the Browse button to navigate to the data source of choice whether it be accessible to your local machine or through the metadata folders available to you. Once identified, clicking the Filter & Sort button brings up a dialog, very familiar to SAS Enterprise Guide users, to identify the variables required and any filtering or sorting activity that must take place before the data is brought into Excel. AMO regulates the number of observations and columns retrieved into Excel to ensure the version-specific row / column limits are respected.

If **PivotTable** is selected, the Excel PivotTtable wizard is displayed, permitting the dataset variables to be dragged into the pivot table area as required. Excel pivot tables can be set to “refresh on open” thus always providing up to date summary information, drawn directly from the back-end data source – no matter where in the enterprise it may have originated from.

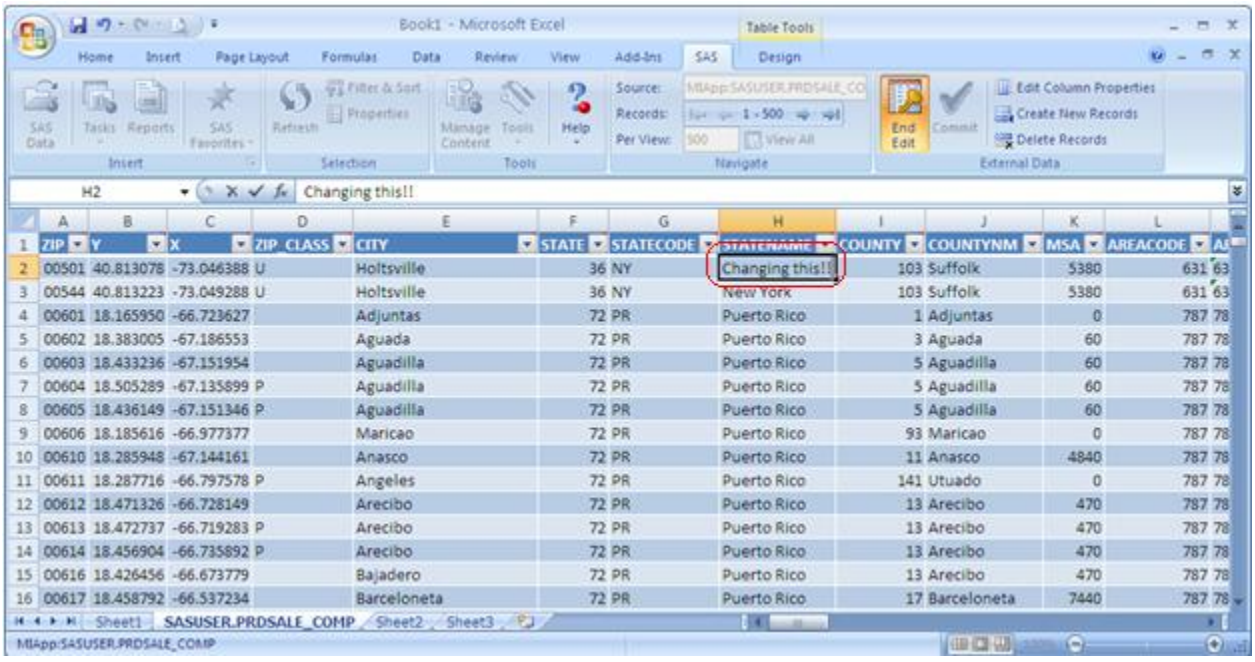

class.xlsx - Microsoft Excel
PivotTable Tools
Home
Insert
Page La
Formula
Data
Review
View
Add-Ins
SAS
Options
Design
SAS Data
Tasks
Reports
SAS Favorites
Insert
Refresh
Filter & Sort
Properties
Selection
Manage Content
Tools
Help
Tools
Source:
Records:
Per View:
A3
MIApp:SASUSER.PRD
To build a report, choose fields from the PivotTable Field List
PivotTable Field List
Choose fields to add to report:
ALIAS_CITY
AREACODE
AREACODES
CITY
COUNTY
COUNTYNM
DST
GMTOFFSET
MSA
PONAME
STATE
STATECODE
STATENAME
TIMEZONE
X
Y
ZIP
Report Filter
Column Labels
Row Labels
Values
Defer Lay...
Update
SASUSER.
Ready
100%

Detail data retrieved into Excel:



Note the **Begin Edit** and **Commit** ribbon choices in the image above. Where the user has the requisite permissions, AMO can be used to create, delete or edit data rows in place – even in tables from remote DB systems.

Begin Edit was clicked, the cell modified and **Commit** checked. The modified cell has been outlined in red.



CONCLUSION

SAS provides MANY different methods of moving data and results between SAS and Excel. Each of them with differing levels of flexibility and utility, but they all get the job done. Many with options that fine-tune the process and deliver just the results you need, others with the bare necessities to get the job done.

Due to additional license requirements, some options may not be available in every installation. The chart below summarizes the capabilities and the requirements of each method.

Description	To SAS	To Excel	Requirements	Pros	Cons
Delimited Text file	Y	Y	Base SAS, PROC IMPORT / EXPORT, data step	No additional licensing required	No formatting
Native Excel files	Y	Y	SAS Access for PC Files license, PROC IMPORT / EXPORT, Windows machine or PC Files Server (for < 9.3)	Familiar functionality	SAS Access license, confusing array of options, may require post processing of data
			Excel Libname	Customized, flexible results	SAS Access license
			SAS Enterprise Guide	No additional licensing required, wizards	
ODS		Y	EXCEL	native Excel files **with graphics**, can be IMPORTed by SAS, Windows not required, no additional license required	NONE !!
			Base SAS, tagsets,ExcelXP, HTML	No additional licensing required, customized results	File sizes can be large
			CSV	No additional licensing required	No formatting
DDE	Y	Y	Base SAS, Windows machine	No additional licensing required, very powerful	Difficult to debug, depreciated functionality
AMO	Y	Y	SAS (Enterprise) Business Intelligence Suite - open data directly in Excel or Pivot - surface results of Stored Processes	Very powerful and flexible functionality, including pivot tables	SAS BI license required

REFERENCES

The paper provides a number ofquotes from SAS Online Documentation. The home page for that documentation is cited below.

SAS Institute, “SAS Online Product Documentation”, 2014-03-12, http://support.sas.com/documentation/onlinedoc/base/index.html#base94

RECOMMENDED READING

There are a number of very worthwhile SAS ← → Excel resources on the internet. The following is a partial list.

DDE

http://en.wikipedia.org/wiki/Dynamic_Data_Exchange

http://support.sas.com/documentation/cdl/en/hostwin/63285/HTML/default/viewer.htm#ddeexamples.htm

http://office.microsoft.com/en-us/excel-help/excel-4-0-macro-functions-HP001047533.aspx

PROC IMPORT / EXPORT Options

http://support.sas.com/documentation/cdl/en/acpcref/63184/HTML/default/viewer.htm#a003103761.htm

Supported Data Sources and Environments:

http://support.sas.com/documentation/cdl/en/acpcref/67382/HTML/default/viewer.htm#n08xs35g2w8mt1n1dhiyyd898s6z.htm

Excel Libname

http://www2.sas.com/proceedings/sugi31/024-31.pdf

http://www.stratia.ca/papers/excel_libname.pdf

ODS Tagsets

http://support.sas.com/resources/papers/proceedings11/250-2011.pdf

http://www.nesug.org/proceedings/nesug08/ap/ap06.pdf

http://www.sas.com/events/cm/867226/ExcelXPPaperIndex.pdf

http://support.sas.com/rnd/base/ods/odsmarkup/msoffice2k/index.html

ODS Excel

http://support.sas.com/resources/papers/proceedings16/SAS5642-2016.pdf

http://blogs.sas.com/content/sasdummy/2014/08/29/experimenting-with-ods-excel-to-create-spreadsheets-from-sas/

SAS Enterprise Guide

http://blogs.sas.com/content/sasdummy/2010/10/01/export-to-excel-20072010-using-sas-enterprise-guide/

CONTACT INFORMATION

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

Name:	Harry Droogendyk
Enterprise:	Stratia Consulting Inc.
E-mail:	conf@stratia.ca
Web:	www.stratia.ca