

Methodologies for converting Excel Spreadsheets to SAS® datasets

Karin LaPann
ViroPharma Incorporated

Much functionality has been added to the SAS to Excel procedures in SAS v.9. In SAS version 8.2 there were some improvements, but one still had to do one of the following to make the Excel sheet readable:

- A) Save it as a “Microsoft Excel 5.0/95 Workbook (*.xls)”, with only one sheet per workbook.
- B) Save it as a comma delimited file “CSV (Comma delimited) (*.csv)” and read in with input statement in a data step.
- C) Save it as text file “Text (Tab delimited) (*.txt)” and read in with input statements. This last form being readable only if you had no missing data in any columns.

The first one could be read directly by SAS using proc import. However, the Excel spreadsheet could only have one tab, ergo the reason to save as Excel 5.0. The second and third options could be read in with data step input statements. The third option was readable only if you had no missing data in any columns, as SAS assumes variables in sequential rather than positional order.

Excel Formatting issues

Other dilemmas caused by Excel setup include the following:

1. Spacing used by Excel users to make spreadsheets readable
2. Titles, footnotes on Excel documents
3. Formulas or macros
4. Dates and times
5. Mixed character and numeric entries in one column
6. Awkward titles across the top, which become variable names

The following Excel sheet will cause some problems.

Labcorp.xls

A	B	C	D	E	F	G	H
1	Labcorp						
2	Austin,						
3	TX						
4	Patient			lab			
5	Id	lab test	lab date	time	value	unit	comments
6	321	Cholesterol	25Jan2006	6:15	160	mg/dL	on statin drug
7	321	Calcium	25Jan2006	6:15	80	mg/dL	
8	321	Bun	25Jan2006	6:15			
9	321	Albumin	25Jan2006	6:15	4	g/dL	
10	321	Basos		6:15	0.5	%	
11	321	abs basos		6:15	0	th/mm3	
12	321	Color Urin		6:15	Yellow		
13		Cannabinoid					
14	321	Urin		6:15	Negative		
14	321	Lymphs	25Jan2006	6:15			

Item 1:
Blank row will cause problems identifying char or numeric

Item 2:
Date field, do you want a date string or a character date?

Item 3:
Comments are allowed up to 200 Characters in SAS. How to

Item 4:
Alpha-numeric data in what originally appears to be a numeric column. SAS will have assigned this column as numeric. Therefore, these will be assigned to missing.

Some possible solutions:

Item 1:

- a. If you have access to the Excel spreadsheet and are allowed to manipulate it, simply delete the offending blank row.
- b. Otherwise, read in using Input starting on line 6, using option GETNAMES = NO;

Item 2:

Dates read in rather nicely to the SAS dataset as dates. However, if the date has been entered on the Excel spreadsheet as a character string, you need to specify in the input statement as a character string, then convert to date as follows:

```
labdt = input(lab_date,date9.); format labdt date9.;
```

Item 3:

Use LENGTH statement to define as Char 200 prior to reading in.

Item 4:

- a. If you have access to the Excel spreadsheet and are allowed to manipulate it, format entire column as character prior to reading in both character and numeric entries. After creation of the SAS dataset, save as separate variables if you need the numeric entries separated.
- b. Otherwise, assign as character field programmatically prior to reading in to a SAS dataset.

Below is a table showing Excel to SAS and SAS to Excel conversions *:

Default SAS Variable and Type Formats for Excel Formats		
Excel Column Format	SAS Variable Format	SAS Variable Type
Text	\$w.	character
General , Number, Scientific, Percentage, Fraction	See Note 3	numeric
Currency, Accounting	DOLLAR21.2	numeric
Date, Datetime, Time	DATE9. See Notes 1 and 2	numeric
<p>1 The default format is DATE9. However, you can use the SASDATEFMT option to change the format to other date or datetime formats. The LIBNAME engine automatically converts the internal date value for you.</p> <p>2 If you have a time only field in your Microsoft Excel range, you can use SASDATEFMT to assign it with the SAS TIME. format. Note that the SAS date/time value uses 01Jan1960 as a cutoff line while the Jet provider date/time value uses 30Dec1899 as a cutoff line.</p> <p>3 To access Fraction or Percent format data in your Excel file, you can use the FORMAT statement to assign the FRACT. or PERCENT. format in your data step code.</p>		

* from SAS v.9.1.3 on-line documentation

Moving to SAS v. 9 (specifically 9.1.3) we have new and exciting ways to read Excel spreadsheets, and also to write back to Excel, in addition to v 8.2 methods.

Some Exciting SAS 9 New Features for use with Excel Conversions

Reading from and Writing to Multiple Sheet Excel Workbooks

The most exciting new feature is the ability to use multiple spreadsheets within one workbook of Excel. We no longer have to save each sheet as its own Microsoft Excel 5.0/95 Workbook. We can now import and export up to and including Excel 2000 spreadsheets.

- Import using the SAS/ACCESS interface for PC Files. (Requires extra license)
- Import Excel spreadsheets (version 5.0 and later) by specifying DBMS=XLS This enables access to Excel spreadsheets on UNIX directly, without going to a PC server.

The Import Procedure

- You can write code:

```
PROC IMPORT  
DATAFILE= "c:\myfiles\testing.xls"  
OUT= data.project101;  
SHEET= "Sheet1" (Note use "Sheet 1$"n if spaces in the name)  
GETNAMES= Yes ; (Note use No and SAS will assign Var0, Var1 etc)  
RUN;
```

- You can use the IMPORT wizard within a SAS Interactive session, then save the generated code and re-use

The Export Procedure

- You can use translation engines (DBMS=XLS) and specify the Excel Workbook version:

```
PROC EXPORT  
DATA= data.project101  
DBMS= Excel2000  
OUTFILE= "c:\myfiles\testing.xls"  
SHEET= "Sheet1" ; (Note use "Sheet 1$"n if spaces in the name)  
RUN;
```

Some Exciting SAS 9 New Features for use with Excel Conversions

The Libname Statement with Excel

The SAS engine now recognizes Excel spreadsheets using the libname command. For this example, the spreadsheet name is CDISCtabs.xls, and the sheets are: VS domain, DM domain, EX domain. The macro variable &sdtmXls refers to a spreadsheet. Following is sample code to access a spreadsheet to get metadata using Excel:

```
%let sdtmXls = CDISCtabs.xls ;

libname sdtmXls Excel "H:\WORKAREA\&sdtmXls"
    access=ReadOnly header=no mixed=yes
    dbgen_name=sas dbmax_text=32767
    DBSASLABEL=COMPAT SCAN_TEXTSIZE= YES scantext=no;

proc contents data=sdtmXls._ALL_
    out=_sdtmALL noprint;
run;
libname sdtmXls clear;
```

For the above example, we create a listing of the contents of the spreadsheet that can then be called in for additional manipulations with INPUT statements.

Now read in or print directly, spreadsheets and also individual Worksheets as follows:

```
libname sdtmXls odbc dsn=Excel;

proc print data=sdtmXls.'DM domain$'n; run;

data mylib.new;
    set mylib.DM domain$'n;
run;
```

source: <http://support.sas.com/kb/12/628.html>

Some Exciting SAS 9 New Features for use with Excel Conversions

Use ODS to HTML format and save as Excel spreadsheet

Here is a simple SAS Dataset which we can convert to HTML format using SAS v.9.1.3. In the ODS HTML command we assign an Excel name and save. The file can now be opened using Excel and has descriptive labels and formatted \$ amounts.

```
Title1 'Report of Sales of Construction Materials';
Footnote1 'PA includes part of DE';
Data sales;
Input date$ 1-10 salesp $ 12- 32 prod $ 34-49 amt region $ 60-61;
Cards;
02/14/2007 David Smith      Block      500,000  PA
02/15/2007 John Doe        Interlock Paver    6,500    NJ
02/16/2007 Jim Jones       Groundface Paver  72,000   NJ
;
```

```
Title1 "Report of Sales of Construction Materials";
Ods html file = "H:\WORKAREA\Sales_report.xls";
Proc report nowindows data = sales;
columns region salesp date prod amt;
  define region /display 'Sales Region';
  define salesp/display 'Sales Person';
  define date /display 'Date of Sale' format=yymmdd10.;
  define prod /display 'Product';
  define amt /display 'Amount' format= dollar11.2;
run;
ods html close;
```

In addition, you can create a spanned header across two or more cells above the table by replacing the following notation in the columns section:

```
columns region salesp date ('Product Information' prod amt) ;
```

Some Creative Uses for uses for SAS and Excel Conversions

Data Entry - Excel is often used by the layperson for rudimentary data entry. Also, it can be quite complex, with the addition of formulas and other reference values. Each cell in Excel has properties which may or may not be transformed to the SAS dataset, depending on the version of SAS you are using.

Lookup Tables – Data can be converted programmatically using lookup tables generated in Excel then output in SAS

For example:

Recode values by using the lookup table.

A	B	C	D	E	F
1	LABPANEL	LABTEXT	LABUNIT	LABSTANDARDUNIT	FACTOR
2	Chemistry	BUN	mg/dL	mg/dL	1
3	Chemistry	BUN	MG/DL	mg/dL	1
4	Hematology	Platelets	X 10 ⁻³ /uL	10 ³ /mm ³	1
5	Hematology	Platelets	K/UL	10 ³ /mm ³	1
6	Hematology	Platelets	X 10 ⁹ TH/L	10 ³ /mm ³	1
7	Hematology	Platelets	X10E + 09/L	10 ³ /mm ³	1

By creating a SAS dataset from the above lookup table maintained in Excel, then merging into our lab dataset we can standardize the units to a standard consistent unit across all labs used in the study.

Tables of Contents – An Excel spreadsheet can be developed to drive titles and footnotes in reports, and even as metadata for automatic creation of tables and listings for pharmaceutical studies.

Reporting for Upper Management - SAS data can ultimately be displayed in Excel Spreadsheets that are then sent to end users who might add graphs and formulas to them. In SAS v. 9 we can even add shading of colors to the cells to make them visually appealing

Some Creative Uses for uses for SAS and Excel Conversions

Maintaining SAS Format Catalogs – Catalogs for Pharmaceutical studies can more easily be maintained in an Excel spreadsheet, then read in for each study. By using names needed by PROC FORMAT, the catalog gets read in smoothly.

Sample Excel spreadsheet format.xls

A	B	C	D	E	F
1	FMTNAME	START	END	LABEL	TYPE
2	YESNOFM	1	1	YES	N
3	YESNOFM	2	2	NO	N
4	YESNOFM	9	9	UNK	N
5	GENDER	M	M	MALE	C
6	GENDER	F	F	FEMALE	C
7					

```
proc access dbms=Excel;
  create work._imex_.access;
  path = &xlspath;
  scantype = yes ;
  getnames = yes;
  mixed = Y;
  create work._imex_.view;
  select all;
run;
```

```
data _fmt;
  set work._imex_;
  if fmtname = '' then delete;
  if end = '' then
    end = start;
run;

proc format cntlin = _fmt library
= library;
run;
```

~~~~~