Paper 024-31

De-Mystifying the SAS® LIBNAME Engine in Microsoft Excel: A Practical Guide

Paul A. Choate, California State Developmental Services
Carol A. Martell, UNC Highway Safety Research Center

ABSTRACT

This paper is a "how-to" guide exploring the SAS v9 Excel libname engine. It focuses on the application of the Excel engine including an introduction and some suggestions on its use. The practical advantages and shortcomings of the engine are discussed. Readers will learn how to use the libname statement to populate and update pivot tables and charts as well as other pre-formatted workbook objects. Essential topics in Excel, such as named ranges are also covered. Techniques in using SAS for renaming, copying, and moving Excel workbooks, worksheets, and ranges will also be mentioned. SAS v9.13 and Excel 2002 and above are used. SAS/ACCESS® software for PC files is required to use the Excel libname engine.

INTRODUCTION

With each new version, SAS offers more ways to get information from the world's best analysis software (SAS®) into the world's most ubiquitous analysis software (MS Excel®). Some popular methods include delimited text files, Proc Export, ODS, and DDE.

DDE is unique because not only can it be used to map data into Excel tables, it can also execute the Excel command processor using Excel v5 macros, enabling much of the functionality of Excel from within SAS. Not only can data be written to a spreadsheet, the spreadsheet can be formatted and used to present information to the end user. As contrasted with the other methods of writing SAS data to Excel, where Excel is simply an empty vessel for SAS to pour information into, DDE dynamically blends the power of the two applications. Unfortunately DDE has several drawbacks: it is based on an outdated macro language, it is syntactically complicated, and it requires the activation of Excel to operate.

With Version 9, SAS now offers the Excel libname engine. The Excel engine covers the middle ground between plain data dropped into Excel via an exported file or ODS HTML and the outmoded yet highly useful DDE. This paper is a brief guide to SAS's Excel engine, how it functions, and offers some suggestions for its use.

While Excel is an excellent tool and a nearly universal platform for sharing information, simple analysis and reporting, its use should be appropriately limited. Excel is not recommended for significant data collection or storage, or performing rigorous statistical analyses. Numerical precision inconsistencies in Excel have been documented and statistical and other functions may generate erroneous results without warning. Please see the *Recommended Readings* for more information.

GETTING STARTED

The basic Excel libname syntax should be familiar to most SAS users:

```
LIBNAME libref <engine-name> <physical-file-name> <libname-options>;
<SAS Code>
LIBNAME libref CLEAR;
```

For example "`LIBNAME WrkBk EXCEL 'My Workbook.xls' VER=2002;`" opens *My Workbook.xls* in the user's default directory. If the file does not exist, SAS will create the workbook. The workbook becomes available in the SAS Explorer window as a library, and all worksheets and named ranges are visible as tables within the library. The CLEAR option closes the workbook connection when finished so the workbook can be opened or moved.

Once opened in SAS, an Excel spreadsheet or named range can be acted on much like a SAS dataset. With certain restrictions it can be referenced for input or output by the data step or procedures. The chief limit is table size. The amount of information that can be stored in a single spreadsheet or named range is 256 columns and 65,535 rows (2**8 x 2**16-1) leaving one row for the column headings. The number of sheets in a workbook is limited by available system memory.

As an example, the following copy procedure copies three files from the help library into the Excel workbook:

```
PROC COPY IN=sashelp OUT=WrkBk;
   SELECT class retail zipcode;
RUN;
```

The SAS data step can also write Excel tables:

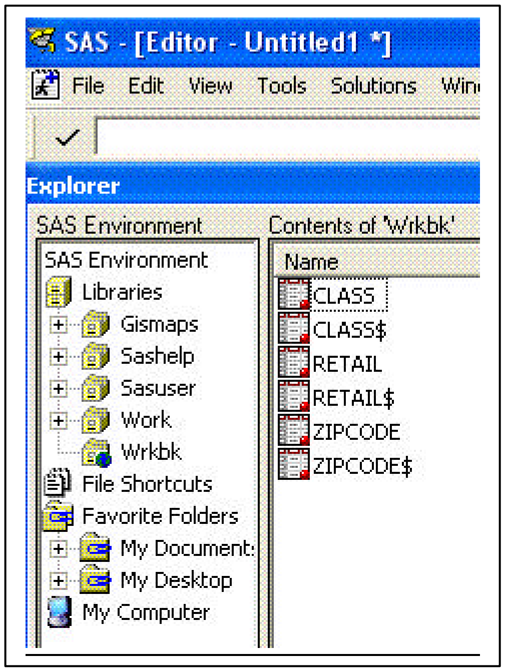
```
DATA WrkBk.class;
     SET sashelp.class;
DATA WrkBk.retail;
     SET sashelp.retail;
DATA WrkBk.zipcode;
     SET sashelp.zipcode;
RUN;
```

The SQL procedure may also used to create Excel worksheets and named ranges:

```
PROC SQL;
  CREATE TABLE WrkBk.class AS
    SELECT * FROM sashelp.class;
  CREATE TABLE WrkBk.retail AS
    SELECT * FROM sashelp.retail;
  CREATE TABLE WrkBk.zipcode AS
    SELECT * FROM sashelp.zipcode;
QUIT;
```

WHAT IS THAT "$" CHARACTER?

Looking at SAS Explorer it may be surprising that each dataset written to Excel appears twice, once with the expected name and once with a trailing "$".



Unlike a typical data source, data in an Excel spreadsheet need not be left and top aligned. For this Excel has *named ranges* which allow data to be placed anywhere inside a spreadsheet. By default SAS reads and writes data from named ranges on spreadsheets, but will also read spreadsheet data directly in the absence of a named range.

When a new SAS dataset is created in an Excel library, SAS creates both a spreadsheet and a named range. Each is given the same name, with the spreadsheet denoted by a trailing "$".

In the example at right *CLASS* is the named range created by the Excel engine and *CLASS$* is the spreadsheet created by the Excel engine to hold the named range. Within SAS, the named range is referred to as *Wrkbk.CLASS*, and the spreadsheet is referenced using the name literal *Wrkbk.'CLASS$'n*.

SAS name literals are name tokens written as strings within quotation marks, followed by the letter n. Name literals allow the use of special characters that are not otherwise allowed in SAS names, like the "$" used by the Excel libname engine to distinguish worksheets from named ranges. For more information see the *Recommended Readings*.

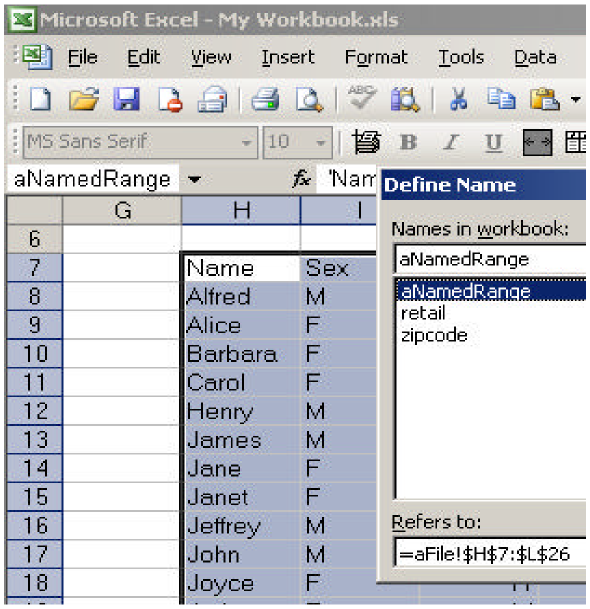
SAS datasets are copied into named ranges on spreadsheets. Thus, using SAS and EXCEL, there are four ways to access the same set of information:

Excel:	Excel Named Range
	Excel Spreadsheet
SAS:	SAS Libname reference of the Excel named range
	SAS Libname reference of the Excel Spreadsheet

In Excel		In SAS	
NAMED RANGE	**WORKSHEET!CELL REFERENCE**	**Named Range Dataset**	**Spreadsheet Dataset**
CLASS	CLASS!$A$1:$E$20	CLASS	CLASS$
RETAIL	RETAIL!$A$1:$E$59	RETAIL	RETAIL$
ZIPCODE	ZIPCODE!$A$1:$P$41989	ZIPCODE	ZIPCODE$

EXCEL NAMED RANGE BASICS

A named range is an Excel tool used to define a range of data or function on a range of data. A named range of data may be defined absolutely or relatively. SAS always creates absolute named range data references. In the above example the named range *CLASS* is defined by SAS within the Excel workbook as *CLASS!$A$1:$E$20*. Named ranges may be created this way in SAS, or they may be defined directly within Excel to be subsequently used by SAS. If defined first in Excel they need not be top-left aligned in cell *A1*.


Microsoft Excel - My Workbook.xls
File Edit View Insert Format Tools Data
MS Sans Serif
10
aNamedRange
'Nam
G
H
I
Define Name
Names in workbook:
Refers to:

	G	H	I
6			
7		Name	Sex
8		Alfred	M
9		Alice	F
10		Barbara	F
11		Carol	F
12		Henry	M
13		James	M
14		Jane	F
15		Janet	F
16		Jeffrey	M
17		John	M
18		Joyce	F

Names in workbook: aNamedRange

aNamedRange
retail
zipcode

Refers to: =aFile!$H$7:$L$26

In Excel there are two methods to define a named range. The user can select a cell range with the mouse or cursor and type a named range name in the *name box* on the top left of the spreadsheet frame. Alternatively, the range can be selected with the mouse or cursor and then under the menu item <Insert> <Name> <Define> the name can be declared. Either method may be used to predefine a placeholder to write a future SAS dataset.

In the example at right the *CLASS* data range is moved to aFile!$H$7:$L$26 and the named range *aNamedRange* is defined.

Once defined, when a named range is selected from the name box drop down, Excel automatically highlights the complete range. The range can then be repositioned with the mouse. Alternatively, the <Insert> <Name> <Define> menu can be used to reposition a named range, although this would not move the corresponding data.

WHAT CAN SAS DO VIA THE EXCEL ENGINE?

The SAS libname engine has the capacity to:

- Create new workbooks.
- Create a new spreadsheet with a named range and write data to the range on the spreadsheet.
- Write data to an empty existing named range.
- Append data to spreadsheet data or named range data.
- Read data from a pre-existing spreadsheet.
- Read data from a pre-existing named range.
- Delete all the data from a spreadsheet.
- Delete all data from a named range.
- Do the above without Excel on the PC.

The SAS libname engine does not have the capacity to:

- Rename spreadsheets within a workbook.
- Delete spreadsheets from a workbook.
- Delete workbooks as a whole.
- Change or apply formatting.
- Delete cells containing formulas.
- Write a formula into a cell.

Note that DDE may be used to create, delete, or rename spreadsheets, as well as to format worksheets and enter and delete formulas. Excel needs to be activated by SAS to process DDE commands, whereas the Excel Libname does not need Excel present or activated.

Filename Pipe or X commands may be used with the operating system to copy or delete workbooks. Workbooks may also be copied with data step Infile and File statements. SAS Filename FTP can be used to email Excel reports.

EXCEL LIBNAME ENGINE BASIC OPERATIONS

ACCESSING DATA FROM A SPREADSHEET

An existing spreadsheet with the data aligned in the top-left A1 cell is read directly using the trailing "$" spreadsheet name convention:

```
LIBNAME WrkBk EXCEL 'My Workbook.xls' MIXED=YES;
DATA work.aFile;
     SET WrkBk.'aFile$'n;
RUN;
LIBNAME WrkBk CLEAR;
```

This copies the spreadsheet data from the sheet *aFile* to the SAS dataset. Note that this does not create a named range on the spreadsheet. All data from the spreadsheet is copied.

If the data is not top-left-aligned, a rectangular range is selected by SAS encompassing whatever data exists on the worksheet. The top left cell of the range is used as the first variable name for the SAS dataset, with the cells to the right used to define the column names or an "fn" naming convention if blank or numeric. In the example above, if the named range *aNamedRange* was defined on the spreadsheet *aFile* in the region $H$7:$L$26 and if there was any data on the sheet outside the range, then reading the worksheet *'aFile$'n* would not read the range data correctly. For this reason using named ranges to define data Excel sources is recommended.

Note the libname option "`MIXED=YES`" is used to read mixed character and numeric data as character data. This circumvents common problems with mixed data entry on spreadsheets. Formula results are read from the worksheet (not the actual formulas).

ACCESSING DATA FROM A NAMED RANGE

An existing named range is read directly with the standard dataset name convention. The name of the worksheet does not affect the named range.

```
LIBNAME WrkBk EXCEL 'My Workbook.xls';
DATA work.aNamedRange;
     SET WrkBk.aNamedRange;
RUN;
LIBNAME WrkBk CLEAR;
```

The named range *aNamedRange* in cells $H$7:$L$26 would be read correctly.

ADDING NEW SPREADSHEET AND DATA

A spreadsheet and named range combination are created the same way as a data set.

```
LIBNAME WrkBk EXCEL 'My Workbook.xls' VER=2002;
DATA WrkBk.class;
     SET sashelp.class;
RUN;
LIBNAME WrkBk CLEAR;
```

This creates both the spreadsheet and the named range. The spreadsheet is named *class* in Excel and *WrkBk.'class$'n* in SAS. The range *class* is created in Excel, starting by default in cell *A1*. The Excel range is referenced as the data set *WrkBk.class* in SAS.

To write data beginning elsewhere than default cell *A1*, first specify an empty named range in Excel. When writing to that range SAS will begin in the top-left cell of the range and add columns or rows as necessary.

Note that whenever a spreadsheet is created with the Excel Libname engine with the VER=2002 option, the corresponding named range is ALWAYS created.

ADDING NEW FILE

If the target workbook file does not exist the Excel libnam e engine will create a new workbook. The default is Excel 97, to specify the most current version use the libname option "VER=2002".

```
LIBNAME WrkBk EXCEL 'My Workbook.xls' VER=2002;
DATA WrkBk.class;
     SET sashelp.class;
RUN;
LIBNAME WrkBk CLEAR;
```

This creates a new 2002 version workbook file containing the spreadsheet, which in turn contains the named range.

Note that using earlier versions of Excel workbooks may disable some functionality in Excel.

REPLACING NAMED RANGE OR SPREADSHEET DATA

The Excel Libname engine cannot replace Excel data by simply overwriting. To replace Excel data first erase with Proc Datasets. This does not affect named ranges or worksheets defined in Excel; it only removes the data they contain. (See **Deleting Data** below).

```
LIBNAME WrkBk EXCEL 'My Workbook.xls';
PROC DATASETS LIB=WrkBk;
     DELETE class;
RUN;
QUIT;
DATA WrkBk.class;
     SET sashelp.class;
RUN;
LIBNAME WrkBk CLEAR;
```

APPENDING DATA TO NAMED RANGE

Data sets may be appended to named range data. The named range in Excel is automatically extended by SAS to encompass the additional data.

```
LIBNAME WrkBk EXCEL 'My Workbook.xls' SCAN_TEXT=NO;
PROC APPEND BASE=WrkBk.class
            DATA=sashelp.class;
RUN;
LIBNAME WrkBk CLEAR;
```

Note the libname option "SCAN_TEXT=NO" must be used to append data. The target area for the appended data must be empty or an error will occur.

DELETING DATA FROM AN EXCEL RANGE

To delete the contents of a named range:

```
PROC DATASETS lib=WrkBk;
     DELETE class;
RUN;
QUIT;
```

This will delete all the cell data contents from the range, but not range definitions, border, shading, or other formatting.

Note that any data from overlapping named ranges within the range will be deleted. Cells containing formulas cannot be deleted with Libname Excel, causing an error in the delete statement log.

DELETING DATA FROM AN EXCEL SPREADSHEET

Deleting the data contents of a spreadsheet works the same as ranges, except the SAS spreadsheet name literal must be used due to the special "$" character:

```
PROC DATASETS LIB=WrkBk;
     DELETE 'class$'n;
RUN;
QUIT;
```

EXCEL LIBNAME ENGINE AND DATA SET OPTIONS

The functioning of the Excel libname engine is modified by libname connection options, libname statement options, and dataset options. Some options are available both on the libname and the data set statement. Following are a few important options useful when using Excel. For more usage information please consult the online documentation.

DBLABEL=YES

During output to Excel, SAS variable labels, instead of variable names, are written out to column headers by changing to the data set option "DBLABEL=YES". All labels written in this way must be unique. This allows long variable labels with spaces and special characters to be written from SAS into Excel column headers.

DBSASLABEL=COMPAT

During input from Excel, this option causes names from Excel column headers to be read into SAS variable labels. This is useful for long names and names with spaces and other special characters. SAS variable names are automatically converted from the column headers into SAS name rule compliant names. See VALIDVARNAME below to optionally use spaces and special characters in both SAS variable names and Excel column headers on both input and output from Excel.

DBSASTYPE=(COLUMN-NAME='SAS-DATA-TYPE')

The DBSASTYPE data set option forces an Excel column to be read as a specified type. This is useful if several spreadsheets with the same data are read with the mixed option, but only some sheets have mixed data in a particular column. In this case, use DBSASTYPE=(VarName='CHAR(8)') and the MIXED=YES options in combination to both read the column as $8. and also to retain the mixed character data.

HEADER=NO

If your Excel data does not have a header row with variable names, the HEADER=NO libname option causes SAS to assign the default variable names F1, F2, F3… Otherwise SAS will attempt to use the data in the first row as variable names.

MIXED=YES

When reading data from Excel, SAS scans a column to determine if it is numeric or character, and if character, determine the width. Normally if an initially numeric column has subsequent character data, SAS will make the variable numeric and set the character values to missing. The "MIXED=YES" libname connection option tells SAS to convert such columns to character, thus saving the character entries.

The mixed option forces a read-only mode, to write data back out to a workbook first clear and reassign the libname.

Numeric columns with no character values will be read as numeric. To force a column to be read as a specific format use the DBSASTYPE data set option.

SCAN_TEXT=NO

In order for SAS to write to Excel it is necessary to set "SCAN_TEXT=NO". This libname option default is "YES" to allow SAS to look down the column to determine the appropriate width of SAS character variables.

VALIDVARNAME=ANY

An alternative to DBLABEL and DBSASLABEL is the "VALIDVARNAME" *system* option. Although only available in Base SAS and SAS statistics procedures, the "VALIDVARNAME=ANY" option allows name literals to be used for variable names. This allows SAS to read and write non-standard variable names without the use of labels. Such variables names may begin with or contain special characters. See *VALIDVARNAME* and *SAS Name Literals* in the online doc.

VER=2002

If no workbook exists, SAS will create one in a specified Excel version, defaulting to Excel 97. To retain the most current functionality *VER=2002* is recommended. Named Ranges may not be created by SAS if *VER=2002* is not specified.

BUILDING DYNAMIC REPORTS

By building a report with target named ranges in Excel and then populating them with SAS, an Excel workbook is used as a presentation shell in which to build a dynamic report. Charts and tables may be populated without disturbing their formats. A pivot table or pivot chart report using a named range as a data source may be updated via the Excel libname engine. Excel named ranges with data appended by SAS are automatically extended, so referencing charts and tables in Excel also automatically update.

SAS data are allowable from a single data point to 256 columns and 65,534 rows in size. By appending data, a new line might be added monthly, or a moving window of time might be refreshed. Character and numeric data can be interspersed by defining overlapping named ranges and successively writing, deleting and writing data. Header rows can be defined separately as named ranges and deleted so only data remains. Rows, columns or whole sheets may be hidden within Excel but used as Excel report data sources updated by SAS. Thus, unlimited varieties of Excel reports can be created as shells and then used as a target for SAS to update and distribute.

CONCLUSION

Using the SAS Excel Libname Engine and Excel named ranges, attractively formatted production reports with tables, charts, pivot tables, etc. can easily be built as skeletons in Excel and then recurrently populated by SAS. For much of what complicated DDE code was necessary in earlier versions of SAS, the more simple and direct Libname Excel engine is now available. The Excel Libname engine has many advantages, including familiar SAS syntax and the usual SAS log error reporting. The use of named ranges allows data to be mapped or deleted at any location within a workbook, simplifying the use of Excel as a reporting shell for a SAS source data. The Excel engine also has the added advantage of not executing the Excel command processor in order to function, allowing SAS to update worksheets reports in the absence of Excel.

REFERENCES

SAS 9.1.3 XP Platform
SAS Institute Inc., Cary, NC

SAS OnlineDoc 9.1.3 for the Web
SAS Institute Inc., Cary, NC

Documentation for SAS Products and Solutions
http://support.sas.com/documentation/onlinedoc/index.html

SAS/ACCESS 9.1 Interface to PC Files: Reference
Chapter 2 - The LIBNAME Statement for PC Files on Windows
Chapter 15 - Microsoft Excel XLS Files

The EXPORT Procedure
Data Source Statements; Statements for PC Files, Spreadsheets, or Delimited Files

Rules for Words and Names in the SAS Language
SAS Name Literals

SAS System Options

VALIDVARNAME= System Option

Microsoft Office Excel Professional Edition 2003
Microsoft Corporation, One Microsoft Way, Redmond, WA

RECOMMENDED READING

SAS-L@LISTSERV.UGA.EDU
http://listserv.uga.edu/archives/sas-l.html

Working With Named Ranges in Excel
Pearson Software Consulting, LLC
http://www.cpearson.com/excel/named.htm

Microsoft Excel 2000 and 2003 Faults, Problems, Workarounds and Fixes
Summary: http://www.csdassn.org/reportdetail.cfm?ID=509
Full Text: http://www.daheiser.info/excel/frontpage.html
(Consolidation of criticisms, reported errors and faults in Excel's statistical and other functions.)

Spreadsheet Addiction by Patrick Burns
http://www.burns-stat.com/pages/Tutor/spreadsheet_addiction.html
(Commentary and references on Excel misuse.)

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Feel free to contact the authors at:

Paul Choate, Senior Programmer Analyst
California Department of Developmental Services
1600 9th Street, Sacramento, CA 95818
Work Phone: (916) 654-2160
E-mail: pchoate@dds.ca.gov

Carol A. Martell, Applications Specialist
University of North Carolina Highway Safety Research Center
CB# 3430, Chapel Hill, NC 27599
Work Phone: (919) 962-8713
E-mail: carol_martell@unc.edu

Join the SAS-L @ LISTSERV.UGA.EDU or Google Groups!