Paper FP_82

It's not the Yellow Brick Road but the SAS PC FILES SERVER® will take you Down the LIBNAME PATH= to Using the 64-Bit Excel Workbooks.

William E Benjamin Jr, Owl Computer Consultancy, LLC, Phoenix AZ.

ABSTRACT

SAS users who also use Excel or produce Excel workbooks will eventually find that the rapid pace of hardware and software changes occurring today will soon meet them head on. The need for faster computers and bigger workbooks are accelerating. This issue is not new, 8-Bit computers were replaced with 16-Bit CPU's, and they lost the battle to 32-Bit computers. Today new computer hardware routinely comes with up to eight 64-Bit CPUs in a single 2X2 piece of hardware for an affordable price. The fictional path down the yellow brick road to OZ was fraught with many challenges. SAS Institute has stepped up to the challenge of the way their users mix and match hardware and software. Since SAS Institute could not control the users and their hardware, the institute expanded the way programmers can use the LIBNAME statement to create a PATH= to open the way to create and read new Excel formats. Some new interfaces were built to facilitate transfers, one interface called the PCFILES server allows passing data to and from SAS and Excel across the boundary between 32 and 64-bit computers and software. The examples presented here will clear the fog and open the doorway past the curtain to view the processes available to open a PATH= to become your company's SAS to Excel wizard.

INTRODUCTION

Since the advent of personal computers using 8 bit hardware over 25 years ago many advances have occurred. Central Processing Units (CPU) have gotten more and more powerful, while the chips have remained about the same size and the number of computer circuits in each CPU has increased. Today the basic computer instruction uses 64 bits of information while the first Personal Computers only used 8. Additionally, the same 2 by 2 CPU chip now may hold 4 or 8 CPU's. We are currently transitioning from the 32 Bit computers and software to the new 64 bit computers and software. Now with millions of Personal Computers operating around the world that transition may take some time for the software to catch up with the new hardware. The problem is that the software makers need to be able to provide their product without regard to what hardware the customers have. Or they will not have any customers. SAS Institute and Microsoft both must meet the needs of their customers.

THE PROBLEM

When your boss asks you for the latest analyses of your newest data file – in a spreadsheet, how do you get it there? Over the weekend the IT department just upgraded your computer to a new 64-bit laptop, with 64-bit SAS software and the old 32-bit Microsoft Excel package. (or the other way around) Now a five minute job seems to have no solution.

Even though Microsoft does not seem to care about how other software packages can access their files, SAS has put a lot of time and effort into helping their customers use SAS Software to access other file structures, in this case the much requested Excel spreadsheet. SAS specifically addresses the transfer of data from 32-bit to 64-bit files. An interface program called the SAS PC FILES SERVER® has been designed and implemented to serve this purpose. When SAS software is installed with the SAS PC FILES SERVER active and the Microsoft ACE (Access Database Engine) software installed this problem can be solved.

THE SOLUTION

As with just about anything relating to SAS software there are multiple was to accomplish a specific task. Here we will focus upon one solution. Note that the "Windows" operating system is not as important as the SAS software version, the Excel software, the SAS PC FILES SERVER version or the Microsoft ACE software. Here we will focus on what I have, and since I do not have unlimited resources The O/S may seem a little bit older than the most current versions but the other software working. I am using Windows Vista Ultimate for 64-bit hardware, with Base SAS version 9.3 TS1M0 for a Windows 64-bit system, and Microsoft Excel 2010 (Version 14 – 32-bit). Even though the title says 64-bit Excel The key to making the processing work is to make sure that all of the products are reading and writing to the expected "Bit Alignment" a SAS PC FILES SERVER that runs in 64-bit mode won't run on a 32-bit computer. Contact SAS Technical Support to resolve any issues.

64-bit compatible components	32-bit components
My Hardware	Excel 2010
My Operating System	
My SAS Software	
My SAS PC FILES SERVER	
Microsoft ACE software	

Now the SAS LIBNAME statement PATH= option comes to my rescue.

First let us look at the different syntax options and parameters available for the LIBNAME statement on various operating systems as they relate to accessing Microsoft Excel files. These options are available for versions 9.0 and forward unless otherwise noted. Some general LIBNAME options are discussed, but the main point of the following tables are to point out LIBNAME options specifically designed for use with Microsoft Excel files.

For SAS/Access Interface to PC Files

```
LIBNAME libref <engine> <physical-file-name>
<SAS/ACCESS-engine-connection-options>
<SAS/ACCESS-libname-options>;
```

LIBNAME Syntax Descriptions	
Argument/Alias	**Description**
Libref	A user assigned valid SAS name one to eight characters long
ENGINE	An optional name of a SAS method of accessing a data file. See the documentation for your SAS version to determine if you can access the newer versions of Excel. Microsoft Excel engines include 5, 95, 97, 2000, 2002, 2003, 2007, 2010. There is no engine for Excel4. **SAS Version 9.2 TS Level 2M3 and above allows access to the Microsoft Office 2007 and 2010 products Excel 2007 and Excel 2010**.
physical-file-name	This is the path and file name of a Microsoft Excel file.
SAS/ACCESS-engine-connection-options	See additional resources listed below.
SAS/ACCESS-libname-options	See additional resources listed below.

Figure 1. LIBNAME Syntax Descriptions.

ENGINE-CONNECTION-OPTIONS	
Argument/Alias	**Description**
HEADER= / GETNAMES= or HDR=	Choose to read the first input row from Excel as either variable names or data.
INIT= / INIT_STRING=	Provide an access provider to connect to a data source, see the SAS documentation for further details.
MIXED= / MIXED_DATA= or MIXED_DATATYPE	Choose to allow Excel to determine if data columns contain numeric or character data or cause all data to be read as character data.
PATH= / DATASRC= and DS=.	Identifies the full path and file name of an external Excel file
PROMPT=	Allows for an interactive assignment of a data source.
VERSION= / VER=	This option is used when writing Excel files. The default is to write Excel version 97 (Excel 97 to Excel 2003 format

Figure 2. LIBNAME Engine Connection Option Descriptions.

PATH OPTION

This option allows for the entry of a file name and the full path to access the file. While the same information is entered by using a "`<physical-file-name>`"**,** this option requires the engine name to be included on LIBNAME command.

```
LIBNAME myxls PCFILES SERVER=xxxxx PORT=8621 PATH="c:\demo.xls"; * default port V9.2;

LIBNAME myxls PCFILES SERVER=xxxxx PORT=9621 PATH="c:\demo.xls"; * default port V9.3+;
```

Note that the "PORT=" option is optional when the SAS PC FILES SERVER is installed using the default port. And the "SERVER=" option is not required if the SAS and Excel software are running on the same computer.

EXAMPLES

The rest of the paper is showing examples of methods available to write and read data to and from Excel file. The examples will all use the PATH= option. Some will use a SAS LIBNAME statement, others will use PROC SQL and show you how to access Excel files (read and write) using PROC SQL.

EXAMPLE 1 – WRITING DATA TO A 32-BIT EXCEL FILE FROM 64-BIT SAS

The images on the next page Show a LIBNAME Statement in Figure 1 along with the SAS Code to write a simple dataset to the output Excel file. This code creates an Excel workbook, and expects the SHEET name to ne new. This means that you can write a very large number of new sheets. Be careful because the Microsoft ACE engine may have limitations.
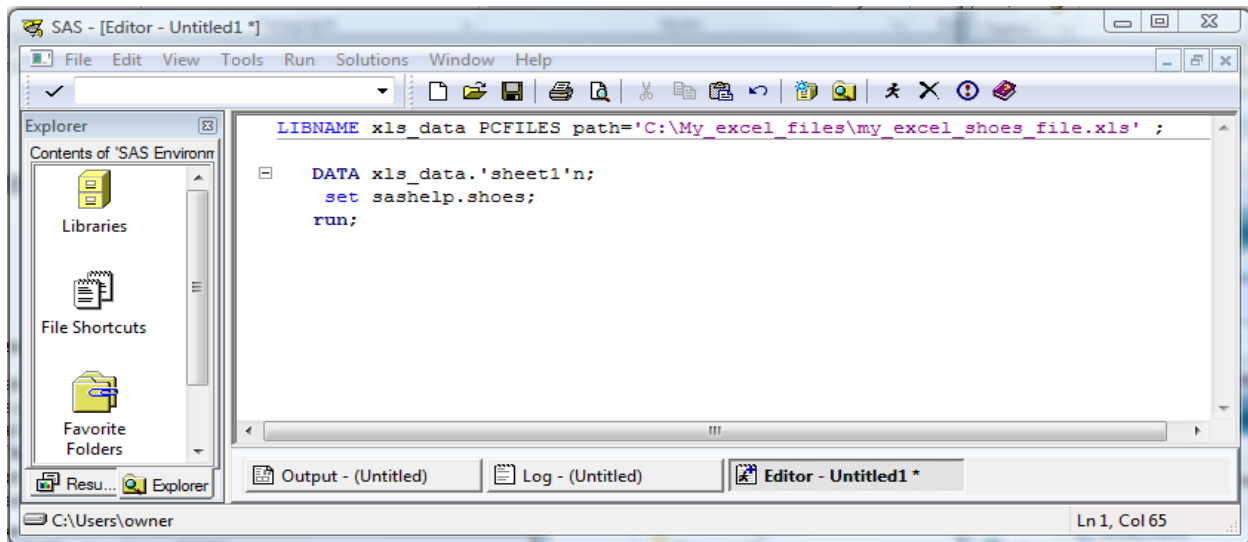
Figure 3. SAS Code to write an Excel file using the SAS PC FILES SERVER.

You may get error messages, but most of the time these are related to the path not existing or the file/sheet existing. Neither of these conditions work well.
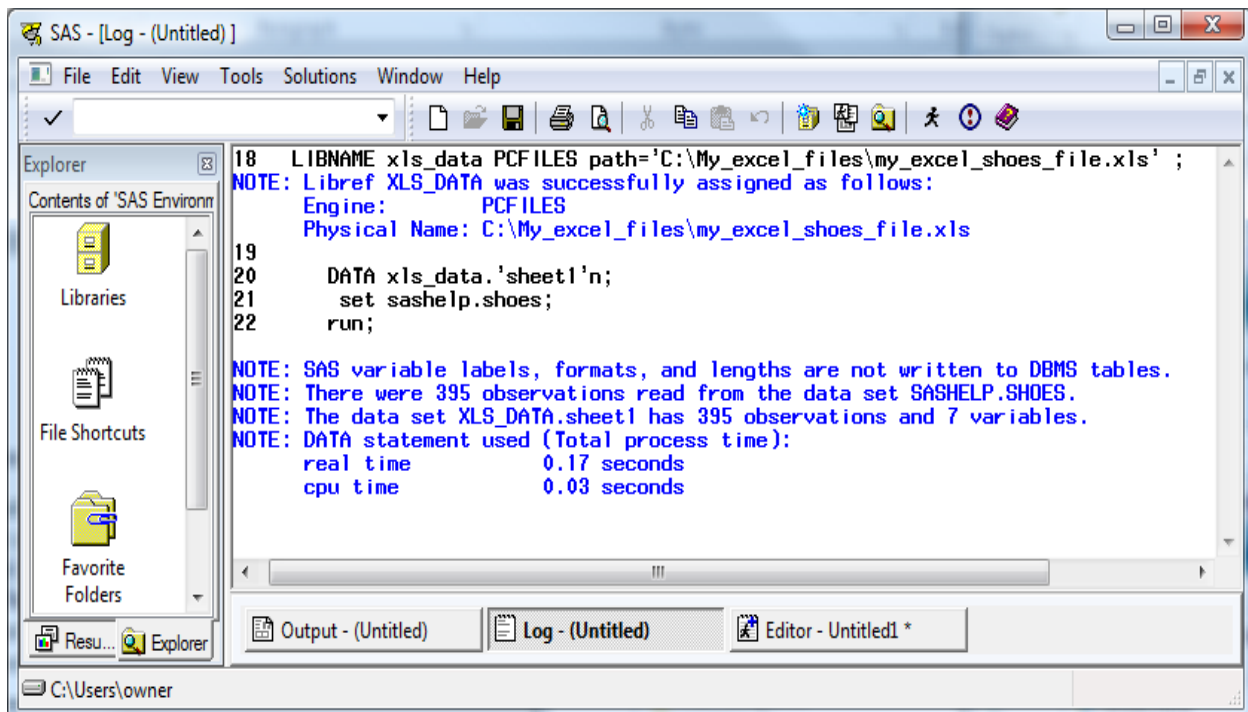
Figure 4. SAS LOG output to write an Excel file using the SAS PC FILES SERVER.

EXAMPLE 2 – READING DATA FROM A 32-BIT EXCEL FILE INTO 64-BIT SAS

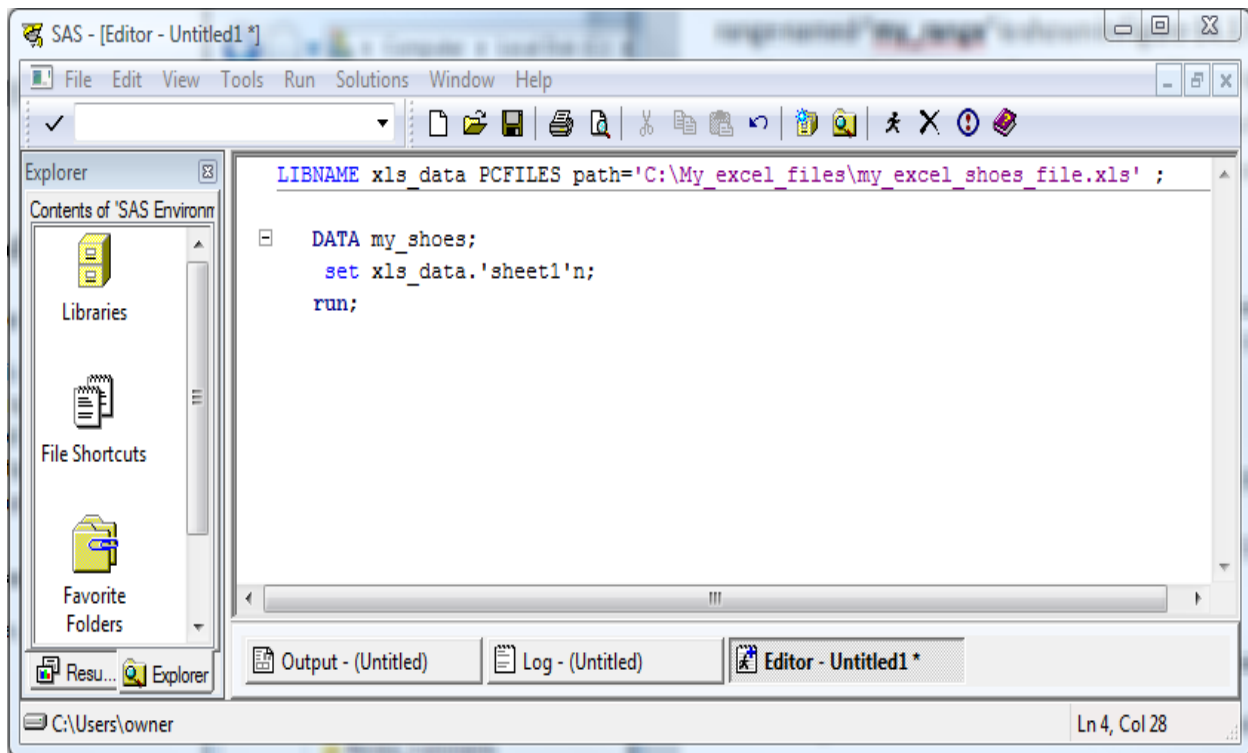The next example shows reading the Excel file just created in example 1.

Figure 5. SAS Code to read an Excel file using the SAS PC FILES SERVER.
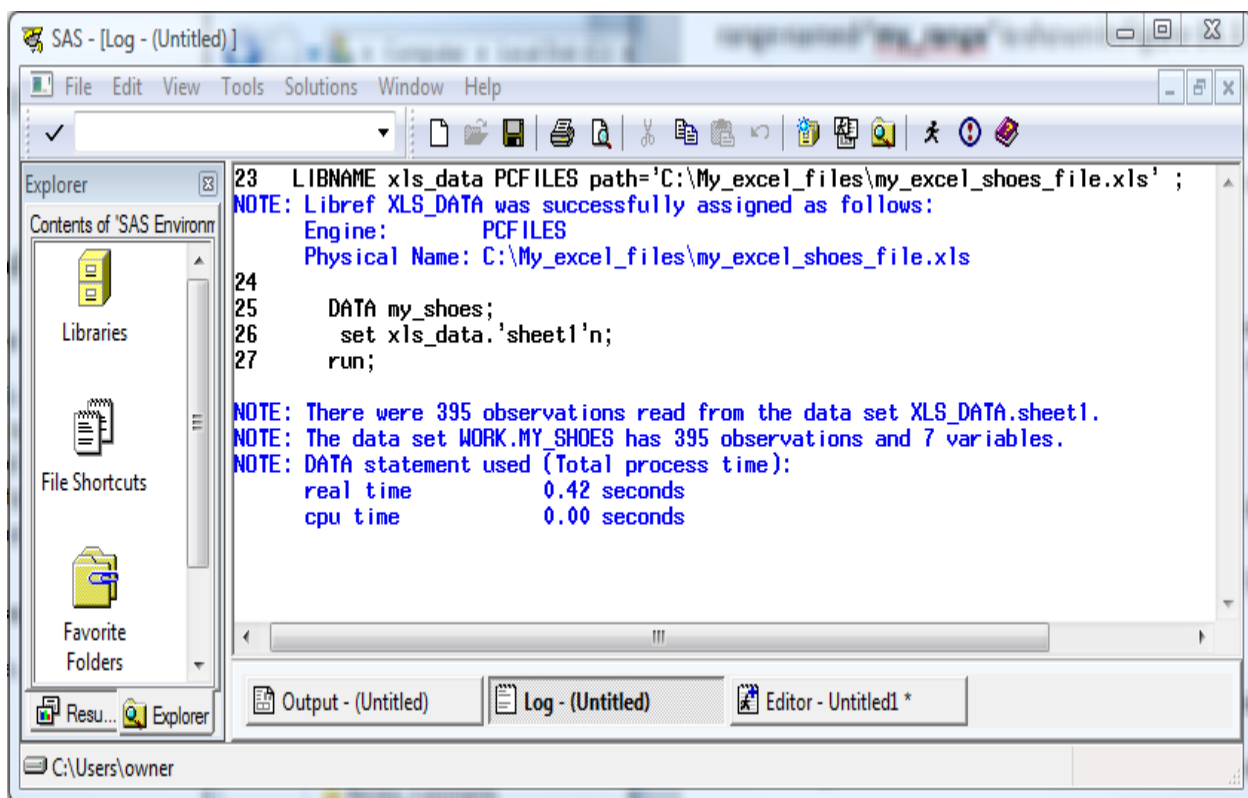
Figure 6. SAS Log output for reading an Excel file using the SAS PC FILES SERVER.

Figure 7. The contents of the SAS dataset created by reading the Excel worksheet.

EXAMPLE 3 – READING DATA TO A 32-BIT EXCEL BINARY WORKBOOK (XLSB)

Figure 8. SAS Code to write an Excel binary file using the SAS PC FILES SERVER.
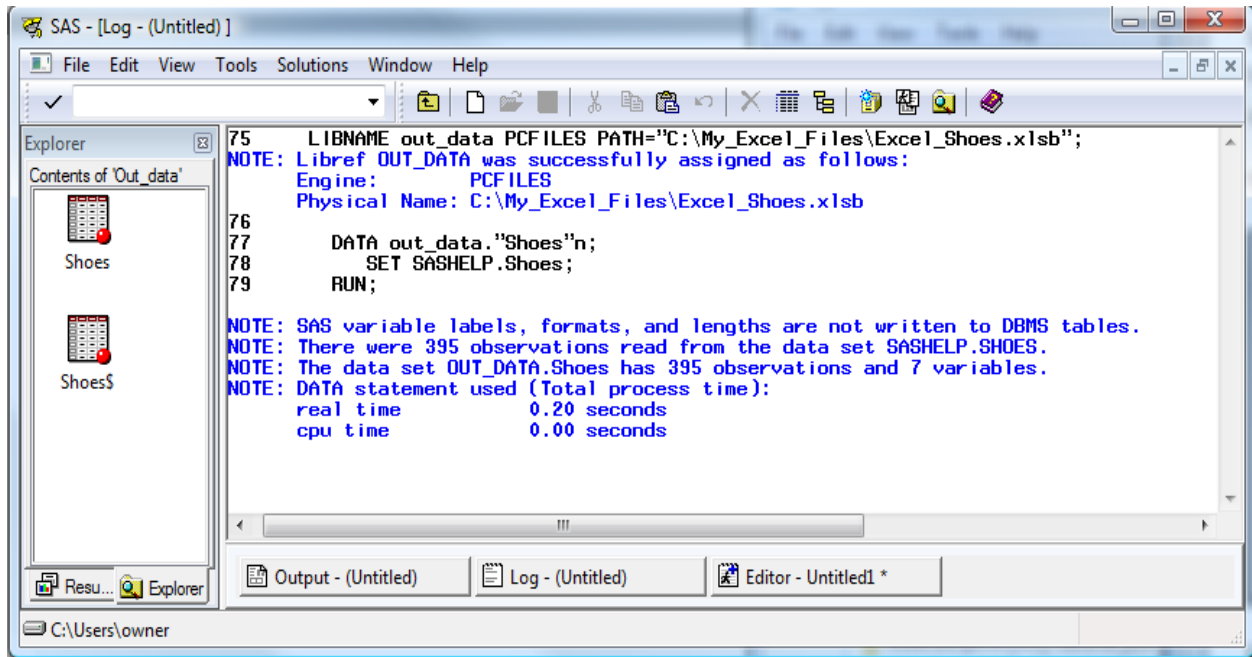
Figure 9. SAS Log output for writing an Excel binary file using the SAS PC FILES SERVER. (XLSB)

Notice that when we create a copy of the workbook and then rename the file with the extension .ZIP we can open the "ZIP" file and view the contents. Most of these files are xml text files but some are in fact binary files. These files are stored in a compressed zip file with the label .xlsb.
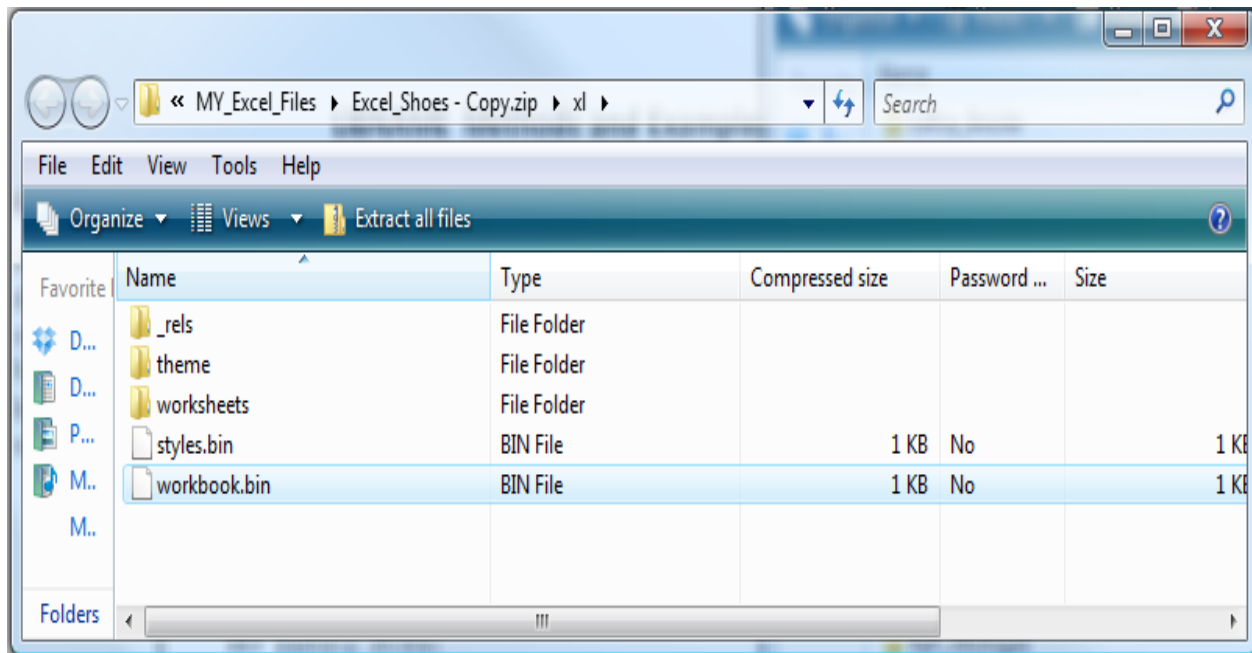
Figure 10. Windows directory listing of the contents of a xlsb file.

EXAMPLE 4 – USING PROC SQL TO READ DATA FROM EXCEL

This example uses PROC SQL to read data from an Excel file. This example just prints the data onto the output window, but with a SQL CREATE command it could just as easily write the data to a SAS dataset.
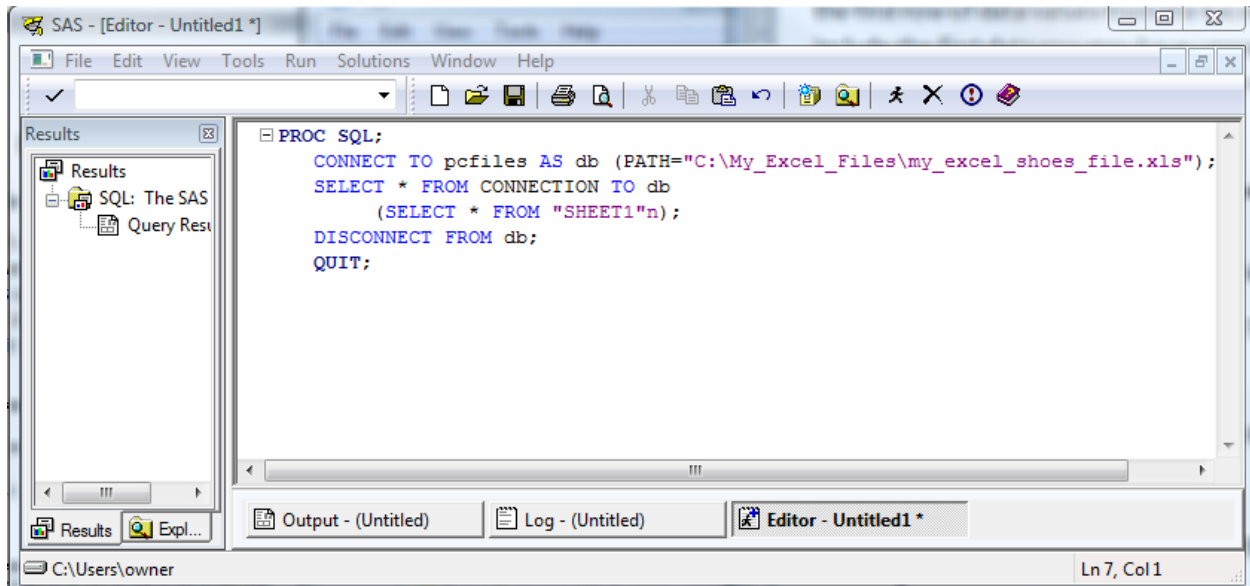
Figure 11. SAS Code to read Excel data using PROC SQL.
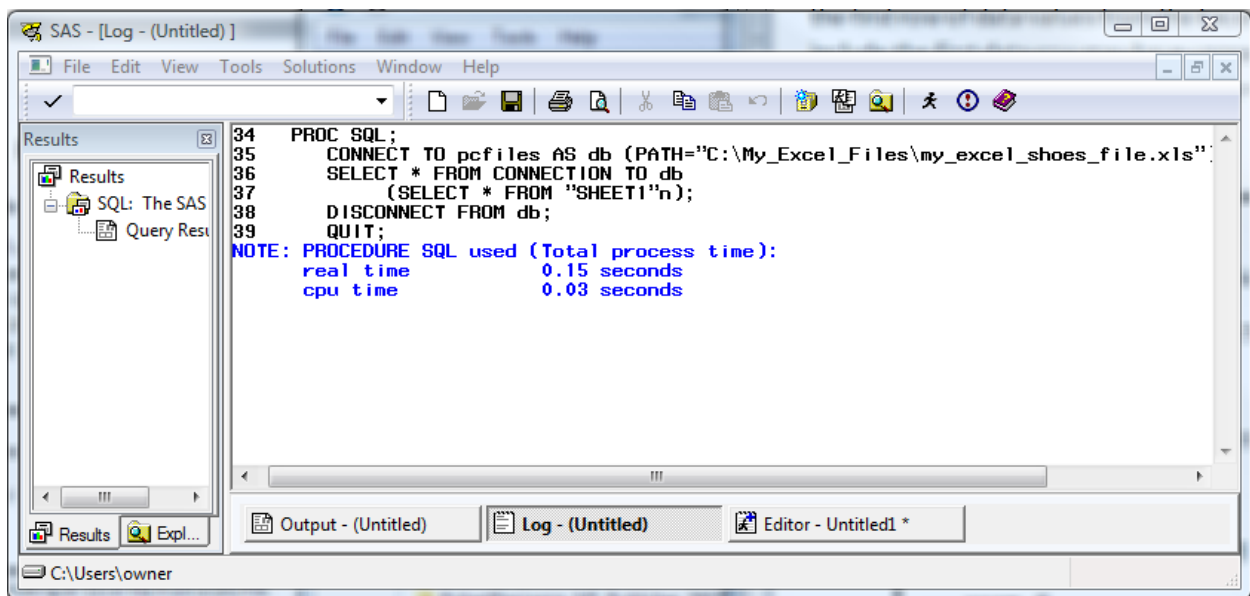
Figure 12. SAS LOG showing the result of reading Excel data using PROC SQL.

Figure 13. The Output Listing generated reading Excel data using PROC SQL.

EXAMPLE 5 – READING DATA FROM AN EXCEL NAMED RANGE INTO SAS
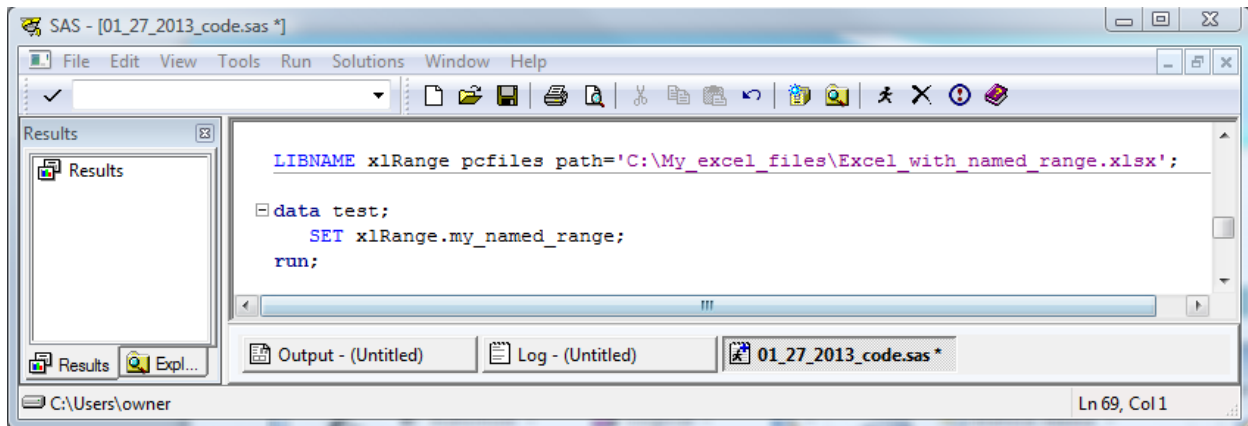
This example shows how to read data from the middle of an Excel Workbook Spreadsheet. When SAS uses a LIBNAME statement to access an Excel workbook it assigns two names to each worksheet. One name is assigned as a "Range" and the other as a "Sheet$". A range name does not have a "$" at the end of the name, while the sheet name does have a "$". When SAS references the data in the Excel workbook a sheet name needs to be referenced as a SAS "Named-Constant" as follows: SET libname.'SHEET1$'n; But, a range name can be referenced as SET libname.range; (without the ' 'n notation).

The screen shot below is of an example of a Named-Range in the middle of an Excel spreadsheet. Many people would first try using DDE to capture data in the middle of a spreadsheet, but if the data can be assigned a named range by Excel than SAS can read the data directly.

Figure 14. An example of an Excel named range.

The SAS code in Figure 15 shows how to read the Excel named range.

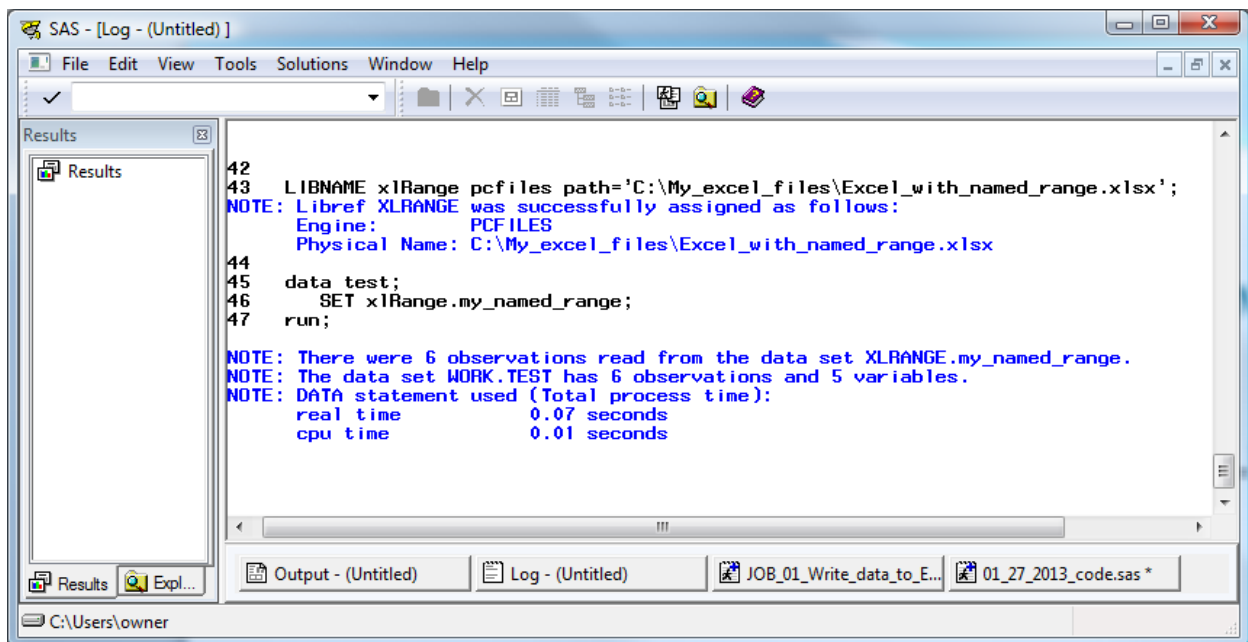Figure 15. SAS code used to read a named range from Excel, NOTE there are no quotes on the SET statement.

Figure 16. SAS Log showing the results of reading a named range from Excel.
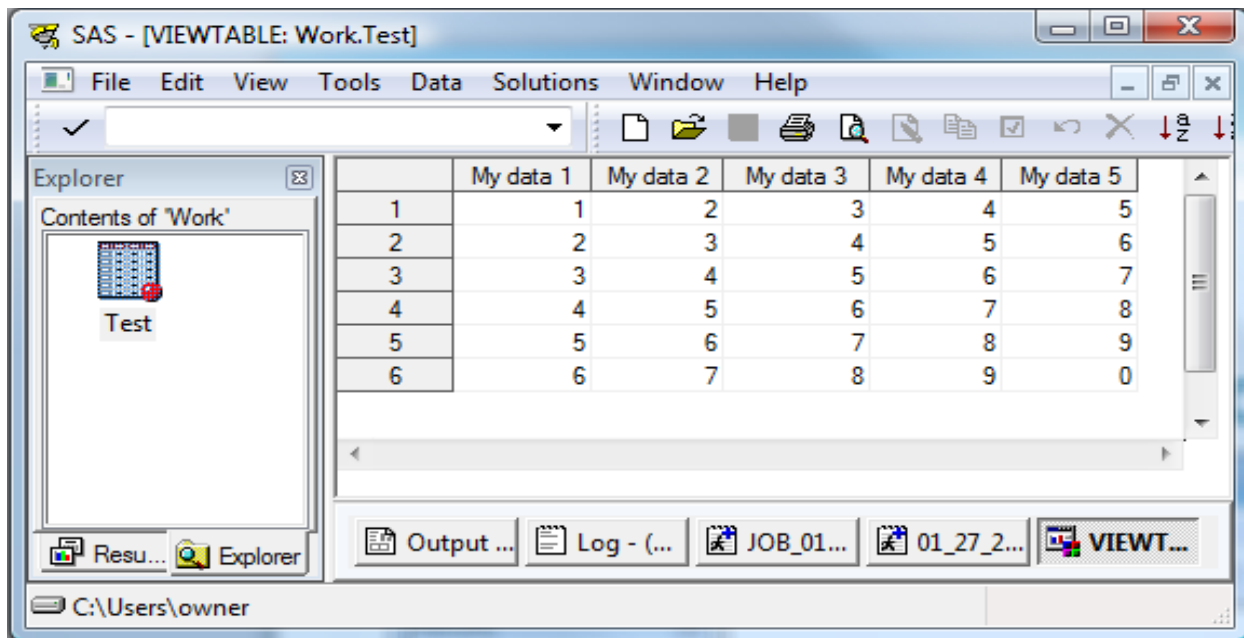
Figure 17. SAS View showing the results of reading a named range from Excel.

Figure 17 shows the results of reading the named range from Excel, the variable labels are shown as evidenced by the spaces visible as "My data 1", "My data 2" etc. The actual variable names have underscores instead of spaces as "My_data_1" etc.

EXAMPLE 6 – SHOWING OUTPUT FILE STRUCTURES

This example is a little different from the other examples. It does the same work, that of writing data to an Excel Workbook. But, the object of this example is to shoe that the same SAS DATA STEP will write out Different Excel WORKBOOK formats. We want to write out a Version 97-2003 Excel workbook, a Version 2007/10 formatted Excel (xlsx) Workbook, and a Version 2007/10 formatted Excel Workbook.
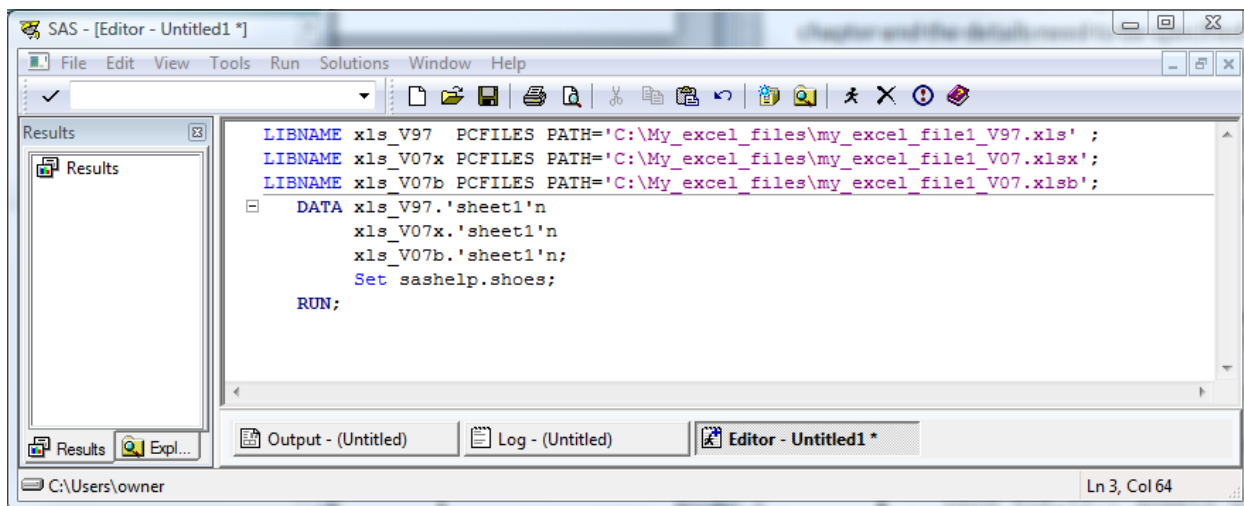

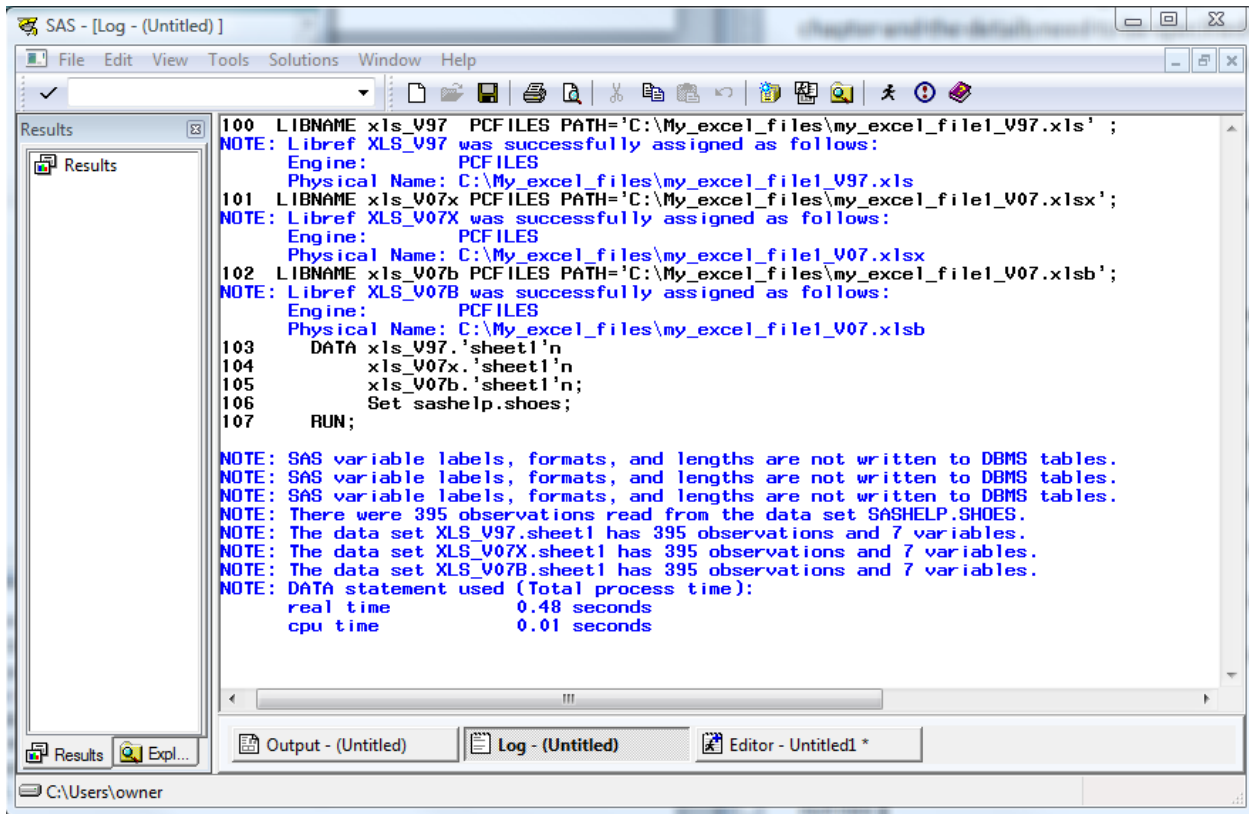Figure 18. SAS Code to write three Excel files of different formats.

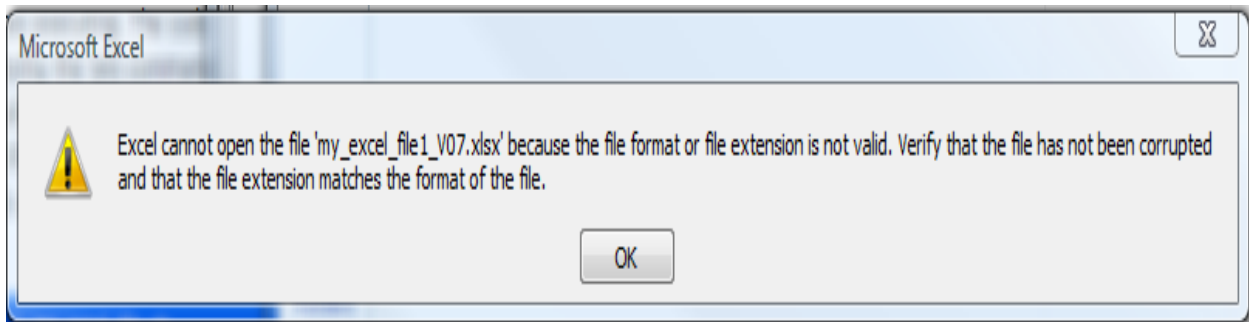Figure 19. SAS LOG showing the successful completion of the job step.

Figure 20. Windows error message displayed when trying to open the xlsx file.

We need to look at this example a little closer, since Excel does not thing this file is acceptable. First let's create a new file directly with Excel 2010. (file EXCEL_2012_Created_Directl…. Below).
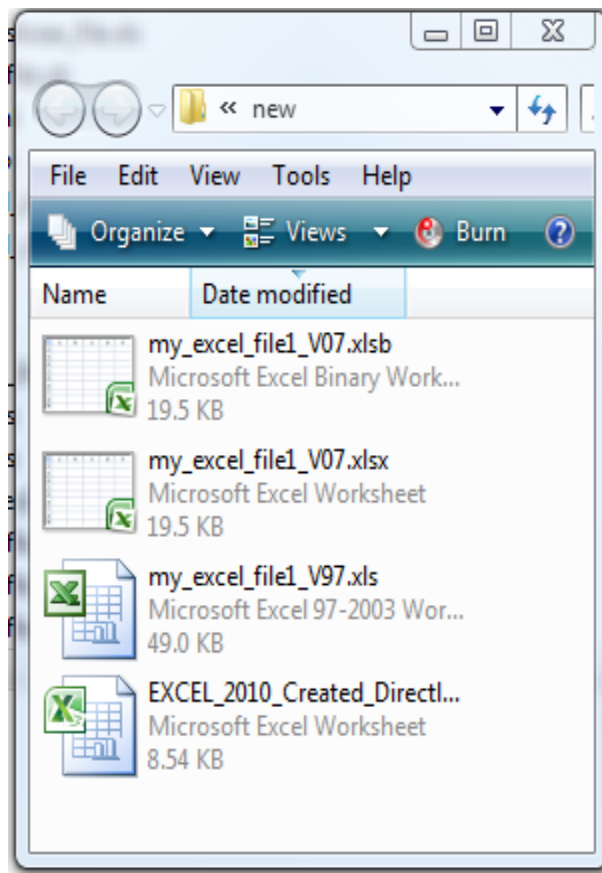
Figure 21. Directly listing of the three files created with SAS and the new Workbook.

One thing nice about Windows icons is that Microsoft goes to great pains to (by default at lease) show different file formats with different icons. In Figure 19 I have displayed the file "Tiles" for each of the four file we just created. Notice that the V97.xls file and the one we created directly are different. The V97 file bears the old Excel icon while the manually created file has the new 2007/2010 icon. Also note that the V07 files both have the same icon. Since we can open the xlsb file (after issuing a LIBNAME statement to clear the LIBNAME references) using Excel that might be a hint about how to fix the problem. If we rename the *.xlsx file to a name that ends in *.xlsb the file should open (if the LIBNAAME has been cleared)

CONCLUSION

The PATH= option when accessing an Excel Workbook using the SAS PC FILES SERVER is a convenient way to open EXCEL Workbooks for either input or output. Unfortunately you need to have the SAS product SAS/Access Interface to PC Files®. But, as with many other products maybe someday you will not need a separate product.

ADDITIONAL RESOURCES

SAS/ACCESS® 9.4 Interface to PC Files Reference

http://support.sas.com/documentation/cdl/en/acpcref/64792/PDF/default/acpcref.pdf

SAS/ACCESS® 9.3 Interface to PC Files Reference

      http://support.sas.com/documentation/cdl/en/acpcref/63181/PDF/default/acpcref.pdf

SAS/ACCESS ® 9.2 Interface to PC Files Reference Second Edition

      http://support.sas.com/documentation/cdl/en/acpcref/63184/PDF/default/acpcref.pdf

SAS/ACCESS ® 9.1 Interface to PC Files Reference

      http://support.sas.com/documentation/onlinedoc/91pdf/index.html

CONTACT INFORMATION

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

Name: William E Benjamin Jr
Enterprise: Owl Computer Consultancy, LLC
Address: P.O.Box 42434
City, State ZIP: Phoenix AZ, 85080
Work Phone: 623-337-0269
E-mail: William@owlcomputerconsultancy.com