Exchanging Data between SAS® and Microsoft Excel
Tips and Techniques to Transfer and Manage Data More Efficiently

William E. Benjamin, Jr.
Contents

Preface ................................................................................................................................. xi
About This Book ................................................................................................................. xiii
About The Author .............................................................................................................. xvii

Chapter 1: Easy Data Movement between SAS and Microsoft Excel ......................... 1
1.1 Introduction .................................................................................................................... 1
1.2 Examination of Excel Files ........................................................................................ 2
  1.2.1 Purpose .................................................................................................................. 2
  1.2.2 Excel Data Types ................................................................................................. 2
  1.2.3 General Excel Workbook Limitations ................................................................. 2
  1.2.4 Excel Workbook Formatting Groups ................................................................. 3
  1.2.5 Excel Data Ranges .............................................................................................. 4
1.3 Examples of Copy-and-Paste Techniques .................................................................. 4
  1.3.1 Highlight, “Cut” or “Copy,” and Then “Paste” ..................................................... 5
  1.3.2 Convert Text Data to Excel Column Data Fields .............................................. 5
  1.3.3 Copy Data to the SAS Enhanced Editor Window for Use in a SAS Program ...... 7
  1.3.4 Save Multiple Lines of Text in a Single Excel Cell ............................................ 8
  1.3.5 Converting Excel Tables to Text ........................................................................ 9
1.4 Accessing Excel Data from the SAS Explorer Window and Toolbar ................. 9
  1.4.1 SAS Explorer Window and Toolbar Processing Method Descriptions .............. 10
  1.4.2 Picking the Export Wizard from the SAS Explorer Window “Export” Menu .... 11
  1.4.3 Using the “Copy Contents to Clipboard” Option of the SAS Explorer Window ... 11
  1.4.4 Selecting the “Save as Html” Option of the SAS Explorer Window ................. 12
  1.4.5 Using the “View in Excel” Option to Copy Data to Excel Files via HTML ......... 13
  1.4.6 SAS Toolbar File Option, the Gateway to the SAS Export / Import Wizards ... 15
  1.4.7 Choosing the “Export Data” SAS Toolbar (Export Wizard) File Option .......... 16
  1.4.8 Electing the “Import Data” SAS Toolbar (Import Wizard) File Option .......... 16
  1.4.9 Using the Export / Import Wizards in a 32/64-Bit Mixed Environment ........... 17
1.5 Chapter Summary ....................................................................................................... 18

Chapter 2: Use PROC EXPORT to Write SAS Data to External Files and Excel
Workbooks ......................................................................................................................... 19
2.1 Introduction .................................................................................................................. 19
2.2 Purpose ....................................................................................................................... 20
2.3 Syntax of the SAS EXPORT Procedure ..................................................................... 20
2.4 Data Access Methods for Excel Files Supported by PROC EXPORT ................. 21
2.5 Overview of the Examples ....................................................................................... 22
2.6 List of Examples

Example 2.1 SAS Code to Export Data to an Excel 4 or Excel 5 Format File ........................................... 23
Example 2.2 PROC EXPORT Using the DBMS=DLM Option ................................................................. 24
Example 2.3 PROC EXPORT Using the DBMS=EXCEL Option .......................................................... 25
Example 2.4 PROC EXPORT Using the DBMS=EXCELCS Option ...................................................... 27
Example 2.5 SAS Code to Export Data to an Excel File with No Column Headers ..................................... 28
Example 2.6 SAS Code to Export Data to a Network Windows Computer ............................................. 28

2.7 Conclusion ........................................................................................................................................... 29

Chapter 3: Use PROC IMPORT to Read External Data Files and Excel Workbooks into SAS

3.1 Introduction ......................................................................................................................................... 31
3.2 Purpose ................................................................................................................................................ 32
3.3 Syntax of the SAS IMPORT Procedure ............................................................................................. 32
3.4 Data Access Methods for Excel Files Supported by PROC IMPORT ................................................. 33
3.5 Overview of the Examples .................................................................................................................. 34
3.6 List of Examples .................................................................................................................................. 34
Example 3.1 PROC IMPORT Using the DBMS=EXCEL4 or EXCEL5 Option ........................................ 35
Example 3.2 PROC IMPORT Using the DBMS=DLM Option .............................................................. 35
Example 3.3 PROC IMPORT Using the DBMS=EXCEL Option .......................................................... 37
Example 3.4 PROC IMPORT Using the DBMS=EXCELCS Option ..................................................... 37
Example 3.5 PROC IMPORT Using the DBMS=XLS or XLSX to Select Columns .............................. 40
Example 3.6 PROC IMPORT Using the DBMS=XLS or XLSX to Select Rows ...................................... 43
Example 3.7 PROC IMPORT Using the DBMS=XLS or XLSX to Select Excel Ranges .......................... 44

3.7 Conclusion ........................................................................................................................................... 45

Chapter 4: Using the SAS LIBNAME to Process Excel Files

4.1 Introduction ......................................................................................................................................... 47
4.2 Purpose ................................................................................................................................................ 48
4.3 Excel-Specific Features of the SAS LIBNAME Statement .............................................................. 48
4.4 Syntax of the SAS LIBNAME Statement ........................................................................................... 49
4.5 LIBNAME Statement ENGINE CONNECTION OPTION Descriptions .......................................... 50
4.5.1 HEADER Option to Read Variable Names ............................................................................... 50
4.5.2 MIXED Option to Select Data Types ....................................................................................... 50
4.5.3 PATH Option to Define Physical File Locations ....................................................................... 51
4.5.4 VERSION Option to Identify Excel File Version ....................................................................... 52
4.5.5 PROMPT Option to Interactively Assign a Libref ................................................................... 52
4.5.6 Other Common SAS PC File LIBNAME Options ..................................................................... 53
4.6 Excel-Specific Dataset Options ......................................................................................................... 53
4.7 UNIX, LINUX, and 64-Bit Windows Connection Options ............................................................ 54
4.8 Overview of the Examples .................................................................................................................. 55
4.8 List of Examples .................................................................................................................................. 55
4.9 Examples .............................................................................................................................................. 56
Example 4.1 Using the Engine Connection HEADER Option .................................................................. 56
Example 4.2 Using the Engine Connection MIXED Option .................................................................. 57
Example 4.3 Using the Engine Connection PATH Option ..................................................................... 58
Example 4.4 Using the Engine Connection VERSION Option ........................................................ 58
Example 4.5 Using Named Literals with the LIBNAME Statement ........................................................... 59
Example 4.6 Using PROC CONTENTS to Examine an Excel Workbook .............................................. 60
Example 4.7 Using Dataset Options to Process Date and Time Values ................................................ 62
Example 4.8 Using Dataset Options to Process Variable Type Conversions ........................................ 63
Example 4.9 Processing on 64-Bit Operating Systems ........................................................................ 64

4.10 Conclusion ................................................................................................................................. 65

Chapter 5: SAS Enterprise Guide Methods and Examples .................................. 67

5.1 Introduction ............................................................................................................................... 67
5.2 Purpose ....................................................................................................................................... 68
5.3 Typical Methods to Access Excel from SAS Enterprise Guide .................................................. 68
5.4 Overview of the Examples ......................................................................................................... 68
5.5 List of Examples ........................................................................................................................ 68
5.6 Examples ..................................................................................................................................... 69

Example 5.1 Using the Export Method with Enterprise Guide ........................................................... 69
Example 5.2 Using the “Send To” Method .......................................................................................... 71
Example 5.3 Using the “Send To” Method to Output a Graph or Report .......................................... 71
Example 5.4 Using the “Export” Method to Output a Graph or Report ............................................... 75
Example 5.5 Using “Open” or “Import” Toolbar Options to Read Excel Workbooks .................... 77
Example 5.6 Using the “Import Data” Toolbar Option to Read a Range of Cells .............................. 80

5.7 Conclusion ............................................................................................................................... 84

Chapter 6: Using JMP to Share Data with Excel ......................................................... 85

6.1 Introduction ............................................................................................................................... 85
6.2 Purpose ....................................................................................................................................... 85
6.3 Methods of Sharing Data between JMP and Excel ...................................................................... 86
6.4 List of Examples ........................................................................................................................ 87
6.5 Examples ..................................................................................................................................... 87

Example 6.1 Within Excel, Set the JMP Preferences for Loading Excel Data ...................................... 87
Example 6.2 Reading Data from Excel to JMP .................................................................................. 88
Example 6.3 Writing Data from JMP to Excel .................................................................................... 89

6.6 Conclusion ............................................................................................................................... 90

Chapter 7: SAS Add-In for Microsoft Office (Excel) ..................................................... 91

7.1 Introduction ............................................................................................................................... 91
7.2 Purpose ....................................................................................................................................... 91
7.3 Methods of Sharing Data Using SAS Add-In for Microsoft Office ............................................ 92
7.4 List of Examples ........................................................................................................................ 94
7.5 Examples ..................................................................................................................................... 94

Example 7.1 Open a SAS Dataset Using SAS Add-In for Microsoft Office ...................................... 94
Example 7.2 Open a SAS Report Dataset (*.srx) Using SAS Add-In for Microsoft Office .................. 99

7.6 Conclusion ............................................................................................................................... 105
Chapter 9: Using OLE DB or ODBC to Access Excel Workbooks................................. 150

9.4 Guidelines for Setting Up OLE DB or ODBC Connections ........................................................ 150
9.5 List of Examples ....................................................................................................................... 150
9.6 Examples ................................................................................................................................ 150
    Example 9.1 Assign a Libref to an Excel Worksheet with the OLE-DB Dialog Box ............... 151
    Example 9.2 Using LIBNAME Prompt Mode to Build an OLE-DB Connection .................... 152
    Example 9.3 Using an OLE-DB init_string to Open an Excel Workbook .............................. 154
    Example 9.4 Using PROC CONTENTS to Verify Excel to OLE DB Connection ................. 154
9.7 Conclusion ............................................................................................................................... 156

Chapter 10: Using PROC SQL to Access Excel Files ......................................................... 157

10.1 Introduction ........................................................................................................................... 157
10.2 Purpose ................................................................................................................................ 158
10.3 Basic Syntax of the SQL Procedure ..................................................................................... 158
10.4 A Simple Explanation of SQL “PASS-THROUGH” Processing ............................................ 160
10.5 Overview of the Examples .................................................................................................... 160
    10.5.1 List of Examples ............................................................................................................ 160
10.6 Examples ................................................................................................................................ 160
    Example 10.1 LIBNAME Assignments to Access Excel Using PROC SQL ............................ 160
    Example 10.2 Create an Excel File, Read It with SQL, and Then Compare the Files ............ 161
    Example 10.3 Use PROC SQL to Read a Subset of Records from an Excel Workbook ........ 162
    Example 10.4 Use PROC SQL Pass-Through Facilities to Process an Excel File ................ 162
    Example 10.5 Read a Pre-defined Range of Cells from an Excel Workbook ....................... 163
    Example 10.6 Calculate a New Variable within the SQL Code and Sort the Output .......... 165
    Example 10.7 Examine the Contents and Structure of an Excel Workbook with a “PCFILES::” Special Query ................................................................. 165
10.7 Conclusion ........................................................................................................................... 166

Chapter 11: Using DDE to Read and Write to Excel Workbooks ......................................... 167

11.1 Introduction .......................................................................................................................... 167
11.2 Purpose ................................................................................................................................ 167
11.3 Basic Concept of the DDE Client-Server Environment ....................................................... 168
    11.3.1 How the DDE Client-Server Relationship Works ........................................................ 168
    11.3.2 General DDE Syntax and Options .............................................................................. 168
11.4 List of User-Written SAS Macros That Can Enhance DDE Processing .............................. 171
    11.4.1 SAS Macro to Start Excel ............................................................................................. 171
    11.4.2 SAS Macro to SAS to Issue Commands to Excel ..................................................... 171
    11.4.3 SAS Macro to Define a Range of Excel Cells for Processing .................................... 172
    11.4.4 SAS Macro to Save the Contents of an Excel Workbook ........................................ 174
    11.4.5 SAS Macro to Close Excel Workbook ...................................................................... 174
    11.4.6 SAS Macro to Write All or Selected Variables to an Excel Output Workbook .......... 175
11.5 List of Examples .................................................................................................................... 177
11.6 Examples ............................................................................................................................. 177
    Example 11.6.1 The Hello World Project ............................................................................ 177
    Example 11.6.2 The Hello World Project When the Excel Workbook Is Closed ................ 179
    Example 11.6.3 The Hello World Project Using NOTAB and LRECL= Options ................. 180
Example 11.6.4 Writing “Hello World” to an Excel File Using DDE Macros ............................... 182
Example 11.6.5 Writing a SAS Dataset to an Excel File Using the SAS_2_EXCEL DDE Macro 184

11.7 Conclusion....................................................................................................................................... 187

Chapter 12: Building a System of Excel Macros Executable by SAS .......................... 189

12.1 Introduction ................................................................................................................................. 189
12.2 Purpose.......................................................................................................................................... 190
12.3 General Design of a Tool to Control Excel Macros from SAS................................. 190
   12.3.1 Prepare a SAS File and Execute Excel to Process the Output ......................................... 191
   12.3.2 Prepare Excel to Open the File Output by SAS ................................................................. 192
   12.3.3 Prepare Excel Macros to Reformat the Excel Workbooks .............................................. 194
12.4 Automate the Tool So That SAS Creates a Formatted Excel Output Workbook ..... 197
   12.4.1 Eliminate the Manual Steps from the Processing............................................................... 197
   12.4.2 Create a SAS Output File with More Data and Control Information ............................... 202
   12.4.3 Create an Excel Macro to Process the Output SAS File ................................................... 203
   12.4.4 Build an Excel Graph Using an Excel Macro ................................................................. 207
12.5 Conclusion....................................................................................................................................... 209

Chapter 13: Building a System of Microsoft Windows Scripts to Control Excel Macros ............................................. 211

13.1 Introduction ................................................................................................................................. 211
13.2 Purpose.......................................................................................................................................... 212
13.3 Guidelines for Building and Using a VBS/VBA Macro Library ................................. 214
   13.3.1 Create Naming Conventions for Storing and Executing VBS/VBA Macros .................... 214
   13.3.2 Set Up Workstation Options............................................................................................. 215
   13.3.3 Where to Store VBS/VBA Scripts and Macros ................................................................. 217
   13.3.4 SAS Code to Execute a Visual Basic Script......................................................................... 219
   13.3.5 Build a Parameter-Driven VBS Script to Control the Execution of Excel .................... 220
   13.3.6 Build a Control Macro for Each Excel Report ................................................................. 223
13.4 Conclusion....................................................................................................................................... 229

Chapter 14: Create an Excel Workbook That Runs SAS Programs ......................... 231

14.1 Introduction ................................................................................................................................. 231
14.2 Purpose.......................................................................................................................................... 232
14.3 Guidelines for Building an Excel User Form Interface ......................................................... 233
   14.3.1 Common Excel and Excel User Form Terms................................................................. 233
   14.3.2 Introduction to the Integrated Development Environment (IDE) ............................... 235
   14.3.3 Using the Integrated Development Environment (IDE) Toolbox Menu .................... 236
   14.3.4 Building a Sample Integrated Development Environment (IDE) Menu .................... 237
   14.3.5 Linking the Integrated Development Environment (IDE) Menu and the Data ............ 239
   14.3.6 Storing Control Information in the Excel Workbook Worksheets .............................. 240
   14.3.7 Set Up Control Variables to Access Data Stored in the Workbook ............................ 241
   14.3.8 Learn How to Make the Excel UserForm Execute .......................................................... 245
14.4 Excel VBA Routines to Make the Workbook UserForm Active ............................. 248
   14.4.1 Initialize the User Form....................................................................................................... 248
   14.4.2 Write the User Parameters to a File in a Working Directory .......................................... 253
   14.4.3 Copy Source Program from a Production Directory to the Working Directory ............ 253
14.4.4 Verify the Output Batch File Points to the Correct SAS Run Time Module ............... 254
14.4.5 A Routine to Save the Changes and Exit the Program ........................................... 255
14.4.6 Directory Structure Associated with the Processing ................................................. 255
14.4.7 Common Issues That Might Occur ............................................................................. 257
14.4.8 Prepare a VBA Macro to Process Your Output Report ............................................ 258

14.5 Conclusion ....................................................................................................................... 259

Index ...................................................................................................................................... 261
Chapter 3: Use PROC IMPORT to Read External Data Files and Excel Workbooks into SAS

3.1 Introduction

This chapter builds upon the Chapter 1 explanation and examples of the SAS Import Wizard, and will explain the syntax, usage, and the results that can be generated when using the SAS IMPORT procedure, specifically PROC IMPORT.

PROC IMPORT is a general purpose routine and is able to read data from text files and Excel workbook files which can exist in several different formats. The ability to read files of many formats makes PROC IMPORT extremely useful. The primary focus of this chapter will be upon reading Excel files. However, some examples will show how to read text files with delimiters because Excel can write files with those formats. The syntax of PROC IMPORT will be explained and the options listed below in Table 3.3.1. One important aspect of PROC IMPORT is its ability to interface with an external Data Base Management System (DBMS). PROC IMPORT has a syntax argument called DBMS that makes this option available and permits access to many different input data formats. Options exist to enable the transfer of data between SAS and many other file formats, but because the focus of this work is moving data from SAS to Excel and back, only options relative to Excel will be explored.

Depending upon which operating system and version of SAS you are using, you may be able to read some or all of the following formats. The details for reading these other formats are explained in the SAS documentation.

- Microsoft Access database files
- Microsoft Excel workbook files
- Lotus 1-2-3 spreadsheet files
- Paradox files
- SPSS files
- Stata files
- dBase files
3.2 Purpose

I will discuss the syntax of the SAS IMPORT procedure and point you to the SAS online documentation for your version of SAS in this chapter. There will be several examples to show you how to write the code to use PROC IMPORT and the results that the examples produce. Because not everyone has the latest version of either SAS or Excel installed on his or her computer, I will not restrict my examples to those newest versions. This chapter will show you how to write SAS code to use PROC IMPORT. Because of the size and complexity of some of the reference tables I suggest that you refer to SAS/ACCESS Interface to PC Files: Reference for the version of SAS that you have installed.

3.3 Syntax of the SAS IMPORT Procedure

PROC IMPORT

DATAFILE= <'filename'> | DATATABLE= <’tablename'> (Not used for Microsoft Excel files)
<DBMS> = <data-source-identifier>
<OUT> = <libref.SAS data-set-name> <SAS data-set-option(s)>
<REPLACE>;
<file-format-specific-statements>;

NOTE: Some features relating to Microsoft Excel 2007, 2010, and 2013 for operating systems Microsoft Vista 64 bit, Microsoft Windows 7 and 8, LINUX, and UNIX, may not be available in SAS versions prior to the third maintenance release of SAS 9.2. Other operating systems may not be compatible until later versions of SAS are released. SAS is not supported on some versions of the Microsoft Windows operating system.

Table 3.3.1 provides a high-level definition of the parts of the syntax for PROC IMPORT as listed above. See SAS/ACCESS to PC Files: Reference for more details about PROC EXPORT in the SAS software version you are using.

Table 3.3.1: General Description of PROC IMPORT Syntax Options.

<table>
<thead>
<tr>
<th>Argument / (Alias)</th>
<th>Required</th>
<th>Definition of the Function of the Argument</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTFILE/(FILE)</td>
<td>Yes</td>
<td>Provide the output file name. DATATABLE is not used for Excel files.</td>
</tr>
<tr>
<td>SAS Data Set Options</td>
<td>No</td>
<td>Options like KEEP=, DROP=, RENAME=, WHERE=, and others may be provided.</td>
</tr>
<tr>
<td>OUT=</td>
<td>Yes</td>
<td>Provide the output SAS dataset name.</td>
</tr>
<tr>
<td>DBMS</td>
<td>No</td>
<td>See Tables below for specific options relating to the individual DBMS &lt;identifier&gt; values. Options are based upon the file types being processed and direct the actions of the SAS PROC IMPORT features.</td>
</tr>
<tr>
<td>REPLACE</td>
<td>No</td>
<td>When “REPLACE” is present then SAS will overwrite an existing output file. A new file will be created if the requested file name does not exist.</td>
</tr>
</tbody>
</table>
3.4 Data Access Methods for Excel Files Supported by PROC IMPORT

The data access methods listed in Figure 3.4.1 are used to read data files Excel has the ability to create. Selecting a DBMS mode determines which utility will be used to process the external file to create an output SAS dataset. The input file may be a text file or an Excel spreadsheet. See the documents listed above for more details about the SAS software version you are using. Some of these data access methods (the DBMS=modes) require SAS/ACCESS Interface to PC Files software to function. You must have SAS/ACCESS Interface to PC Files licensed before you can import files directly from some versions Microsoft Excel workbooks. Some features relating to Microsoft Excel 2007, Excel 2010, and Excel 2013 when using Microsoft Windows, LINUX, and UNIX operating systems may not be available in SAS versions prior to the third maintenance release of SAS 9.2. Because the number of SAS, Excel, and operating system versions is large, I once again refer you to the SAS documentation to help you figure out what you have installed.

If you suspect that your SAS and Excel software may have different bit configurations (32 or 64 bit), contact your IT Department.

The DBMS identifiers listed in Table 3.4.1 are relative to the file formats that Microsoft Excel can read or write. The SAS documentation lists other DBMS identifiers that the PROC IMPORT can read. See the SAS documentation for your version of SAS for other options to read file formats available. Different versions of SAS may not be able to read to all of the versions of Excel.

### Table 3.4.1: DBMS Formats Available for Input.

<table>
<thead>
<tr>
<th>DBMS Identifier</th>
<th>SAS/ACCESS Interface to PC Files Required</th>
<th>General Description of the DBMS Output File</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSV</td>
<td>N</td>
<td>Text file with a comma delimiter</td>
</tr>
<tr>
<td>TAB</td>
<td>N</td>
<td>Text file with a tab delimiter</td>
</tr>
<tr>
<td>DLM</td>
<td>N</td>
<td>Text file with a user-defined delimiter</td>
</tr>
<tr>
<td>EXCEL</td>
<td>Y</td>
<td>Excel workbook (2003 xls – 2013 xlsx)</td>
</tr>
<tr>
<td>EXCELCS</td>
<td>Y</td>
<td>Excel workbook (2003 xls – 2007 xlsx) using the SAS PC Files Server</td>
</tr>
<tr>
<td>EXCEL4</td>
<td>Y</td>
<td>Excel workbook using PROC DBLOAD</td>
</tr>
<tr>
<td>EXCEL5</td>
<td>Y</td>
<td>Excel workbook using PROC DBLOAD</td>
</tr>
<tr>
<td>XLS</td>
<td>Y</td>
<td>Excel workbook using file formats prior to Excel 2007 except Excel 4 and Excel 5</td>
</tr>
</tbody>
</table>

Table 3.4.2 lists some information about the input methods available when reading Excel worksheets. Some of these methods have limitations that are smaller than the full capabilities of the Excel version that created them. These restrictions are as a result of using the Microsoft JET or ACE engines to access the Excel workbooks.

### Table 3.4.2: DBMS Input Methods of Accessing Excel Files.

<table>
<thead>
<tr>
<th>Utility</th>
<th>DBMS Model</th>
<th>Excel Version</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXCEL</td>
<td>LIBNAME statement</td>
<td>5, 95, 97, 2000, 2002, 2003, 2007, 2010, 2013</td>
<td>This DBMS option will use the LIBNAME statement. Depending upon your version of SAS and Excel, access may be limited to the first 65,535 rows and 255 columns.</td>
</tr>
<tr>
<td>EXCELCS</td>
<td>SAS PC Files Server</td>
<td>5, 95, 97, 2000, 2002, 2003, 2007, 2010, 2013</td>
<td>This DBMS option will use the SAS PC Files Server. Depending upon your version of SAS and Excel, access may be limited to the first 65,535 rows and 255 columns.</td>
</tr>
</tbody>
</table>
34 Exchanging Data between SAS and Microsoft Excel

<table>
<thead>
<tr>
<th>Utility</th>
<th>DBMS Model</th>
<th>Excel Version</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXCEL4 or EXCEL5</td>
<td>DBLOAD procedure</td>
<td>4, 5, 95</td>
<td>This is supported only on the Microsoft Windows operating systems and is for SAS 6 compatibility.</td>
</tr>
<tr>
<td>XLSX</td>
<td>XLSX format</td>
<td>2007, 2010, and later formats</td>
<td>Some versions of SAS may not support the Chinese, Japanese, or Korean DBCS character sets or *.xlsb Excel files.</td>
</tr>
</tbody>
</table>

### 3.5 Overview of the Examples

The examples in this chapter will cover several but not all of the DBMS options used with PROC IMPORT. I like to group the input processing for PROC IMPORT into general categories within the DBMS options. Furthermore, I feel I must place a caveat onto these groupings because both SAS and Microsoft Excel are mature products that have changed over time. While these categories are generally accurate, your SAS version, Excel version, and computer hardware may not support every DBMS option, and each DBMS option might operate slightly differently depending upon what software you have installed. So make sure you verify what is available to you by looking in the SAS manual that relates to your environment.

- An example retained for backward compatibility with files in the Excel 4 and Excel 5 formats.
- Text file output options like CSV, TAB and DLM do not require SAS/ACCESS Interface to PC Files because the methods read text files.
- Options that read directly from a formatted Excel file.
- LIBNAME options that both use and do not use the SAS PC Files Server.

The options that generate text files will show one example and explain the differences that make the other options work.

### 3.6 List of Examples

Table 3.6.1 is a general description of the functions included in the examples shown in this chapter. Some of the examples here have minor overlaps in the features to show how they interact when additional features are included.

**Table 3.6.1: List of Examples for PROC IMPORT.**

<table>
<thead>
<tr>
<th>Example Number</th>
<th>General Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>PROC IMPORT Using the DBMS=EXCEL4 or EXCEL5 Option. This example is included for backward compatibility with Excel formats Excel 4 and Excel 5, although I would consider it rare to find a computer using this Microsoft Excel software today. The example shows how to read to these old Excel formats.</td>
</tr>
<tr>
<td>3.2</td>
<td>PROC IMPORT Using the DBMS=DLM Option. This example shows how to use a delimiter to separate input values and read the header row of the input file as data. This example is equivalent to DBMS=CSV and DBMS=TAB but allows you to provide your own delimiter.</td>
</tr>
<tr>
<td>3.3</td>
<td>PROC IMPORT Using the DBMS=EXCEL Option. The three parts of this example all read Excel workbooks that do not need the PC Files</td>
</tr>
</tbody>
</table>
### Example Number | General Description
--- | ---
3.4 | **PROC IMPORT Using the DBMS=EXCELCS Option.** This example shows code that was executed on a 64-bit operating system using a 64-bit copy of SAS 9.3 and a 32-bit copy of Microsoft Excel. Since this computer operating system and SAS use a 64-bit configuration but Excel uses a 32-bit configuration, PROC IMPORT requires the use of the SAS PC Files Server. The “CS” part of DBMS=EXCELCS annotates this feature is in use.

3.5 | **PROC IMPORT Using the DBMS=XLS or XLSX to Select Columns.** This example reads an Excel worksheet with no column headers (variable names) in the output Excel worksheet. It also demonstrates that PROC IMPORT will read an Excel sheet name with spaces.

3.6 | **PROC IMPORT Using the DBMS=XLS or XLSX to Select Rows.** Reading Excel data from selected rows of an Excel worksheet.

3.7 | **PROC IMPORT Using the DBMS=XLS or XLSX to Select Excel Ranges.** This example shows you how to use PROC IMPORT to read a range of cells from an Excel worksheet.

---

**Example 3.1 PROC IMPORT Using the DBMS=EXCEL4 or EXCEL5 Option**

The SAS IMPORT procedure maintains the backward compatibility features required to process Excel workbooks in the Excel 4 and Excel 5 formats. This example shows how to write Excel files in those formats. For Excel 4 workbooks the sheet name is the same as the file name (without the .xls) and there is only one sheet in the workbook. For Excel 5 formatted workbooks, the sheet name is “Sheet1”.

*SAS code to import data from an Excel4 file.*

```sas
* there is only one sheet in Excel4 files;
PROC IMPORT
  DATAFILE='C:\My_Files\shoes_to_Excel_4_file.xls'
  DBMS=EXCEL4
  OUT=shoes_from_Excel_4
  REPLACE;
RUN;
```

*SAS code to import data from an Excel 5 file.*

```sas
* SAS code to import data from an Excel 5 file.;
PROC IMPORT
  DATAFILE='C:\My_Files\shoes_to_Excel_5_file.xls'
  DBMS=EXCEL5
  OUT=shoes_from_Excel_5
  REPLACE;
RUN;
```

**Example 3.2 PROC IMPORT Using the DBMS=DLM Option**

Using PROC IMPORT to read delimited files in Base SAS invokes the External File Interface (EFI), and the following code reads in a delimited file with commas as the delimiter from the external file named Shoes.csv in directory c:\My_files. This example uses the DBMS=DLM option with the DELIMITER=’,’ option to select a comma for the delimiter. In addition, it uses the DATAROW=1 and GETNAMES=NO options. These options cause the input SAS file to make the first row from the *.csv file appear as data in the SAS file.

```sas
* SAS code to import data from a delimited file.;
PROC IMPORT
  DATAFILE='C:\My_Files\Shoes.csv'
  DBMS=DLM
  OUT=Shoes
  DATAROW=1
  GETNAMES=NO
  REPLACE;
RUN;
```
NOTE: In Example 2.2 in Chapter 2, the code for PROC EXPORT used the PUTNAMES=NO option to write the ‘c:\My_Files\Shoes.csv’ output file with no variable names in the first row of the file.

The output log listing below shows the External File Interface SAS code created by the “Generated SAS Datsstep” when the PROC IMPORT step above ran. Notice that the input *.csv file did not have a row of headers associated with the data. So, SAS assigned variable names to the input variables (VAR1 to VAR7).

```
PROC IMPORT
   DATAFILE='c:\My_Files\Shoes.txt'
   DBMS=DLM
   OUT=shoes
   REPLACE;

   DELIMITER=',
   DATAROW=1;
   GETNAMES=NO;
   GUESSINGROWS=400;
RUN;
```

**Output 3.1: Listing of the External File Interface Code Generated.**

```
1 2 PROC IMPORT
3   DATAFILE='c:\My_Files\Shoes.txt'
4   DBMS=DLM
5   OUT=shoes
6   REPLACE;
7
8   DELIMITER=',
9   DATAROW=1;
10  GETNAMES=NO;
11  GUESSINGROWS=400;
12 RUN;

13 **********************************************************************
14 * PRODUCT:   SAS
15 * VERSION:   9.4
16 * CREATOR:   External File Interface
17 * DATE:      17FEB14
18 * DESC:      Generated SAS Datastep Code
19 * TEMPLATE SOURCE:  (None Specified.)
20 ************************************************************************/
21 data WORK.SHOES ;
22 %let _EFIERR_ = 0; /* set the ERROR detection macro variable */
23 infile 'c:\My_Files\Shoes.txt' delimiter = ',' MISSOVER DSD lrecl=32767 ;
24     informat VAR1 $25. ;
25     informat VAR2 $14. ;
26     informat VAR3 $12. ;
27     informat VAR4 best32. ;
28     informat VAR5 $12. ;
29     informat VAR6 $12. ;
30     informat VAR7 $9. ;
31     format VAR1 $25. ;
32     format VAR2 $14. ;
33     format VAR3 $12. ;
34     format VAR4 best12. ;
35     format VAR5 $12. ;
36     format VAR6 $12. ;
37     format VAR7 $9. ;
38     input
```
Chapter 3: Use PROC IMPORT to Read External Data Files and Excel Workbooks into SAS

Example 3.3 PROC IMPORT Using the DBMS=EXCEL Option

Example 3.3 – Part 1

The code in parts 1, 2, and 3 of Example 2.3 in Chapter 2 showed how to create an Excel workbook with different numbers of worksheets. The example shows how to create worksheet names with mixed-case letters in the name. However, this method will not write an Excel worksheet with a blank in the sheet name. The following code will read the Excel file and produce a SAS dataset called “Shoes” in the Work directory. Notice that the RANGE= value for the spreadsheet name was in capital letters and ended in a Dollar sign “$”. The spreadsheet name in the “RANGE=” statement did not need to be in uppercase letters.
Example 3.3 – Part 2.
If we want only part of the input Excel file, there are several ways to go about getting just what we want. The following code brings in only a few cells from the input Excel file. Here, we will also suppress the request to pull the variable names from the first row of the input data, since we are pulling data from the middle of the Excel file.

```
PROC IMPORT
  DATAFILE='c:\My_Files\Shoes.xls'
  DBMS=EXCEL
  OUT=shoes
  REPLACE;
  GETNAMES=NO;
  RANGE='shoes$C2:F4';
RUN;
```

This SAS code does that job. The added command “GETNAMES=NO” and the modification of the “RANGE=” operand are the key parts of this SAS code. The SAS output file looks something like the following:

**Figure 3.1: SAS Output from Reading the Excel Range Using Absolute Addressing of Excel Cells.**

Only 12 cells were read from the Excel worksheet called “SHOES” and the SAS variable names were converted to F1, F2, F3, and F4 because the GETNAMES=NO statement suppressed reading any variable names. The “RANGE=” worksheet name value was in lowercase and included the location of the Excel cells to read into the SAS dataset.

Example 3.3 – Part 3
Users of Excel Workbooks have the option of creating subsets of cells in a worksheet that can be called by name; these areas are called Named-Ranges. Figure 3.2 below shows one of these named ranges called “small_range”. The range name was created while running Excel with the workbook Shoes.xls open.
Chapter 3: Use PROC IMPORT to Read External Data Files and Excel Workbooks into SAS

Figure 3.2: An Excel 2013 Worksheet with a Named Range Called “small_range” Highlighted.

The SAS code below shows how to read the data from the Excel named-range called “small_range” into a SAS dataset. Because the GETNAMES=NO option is used, the variable names F1, F2, F3, and F4 that SAS generated are relatively vague variable names; this example will address a way to correct that issue. The DBDSOPTS= option allows you to use other SAS dataset options to change the output SAS dataset while it is being created. The SAS RENAME= dataset option was used here to change the variable names from F1, F2, … to more descriptive variable names. This is done in one pass over the data and makes the output file more useful when PROC IMPORT finishes. You do not need to make another pass over the data to rename the variables. The PROC DATASETS code adds LABEL values to the SAS dataset. The DBMS=EXCEL form of PROC IMPORT does not allow variable labels to be modified on input of the data; therefore, other code is needed to change the variable labels.

```sas
PROC IMPORT
   DATAFILE='c:\My_Files\Shoes.xls'
   DBMS=EXCEL
   OUT=shoes
   REPLACE;
   GETNAMES=NO;
   DBDSOPTS='RENAME=(F1=Subsidiary F2=Stores F3=Sales F4=Inventory)';
   RANGE=small_range;
RUN;
PROC DATASETS LIBRARY=work NOLIST;
   MODIFY shoes;
   LABEL Subsidiary = "Subsidiary"
                   Stores = "Stores"
                   Sales  = "Sales"
                   Inventory = "Inventory";
QUIT;
```
Output 3.1: Listing of the PROC IMPORT Code generated and the PROC DATASETS Listing.

```
1   PROC IMPORT
2       DATAFILE='c:\My_Files\Shoes.xls'
3       DBMS=EXCEL
4       OUT=shoes
5       REPLACE;
6       GETNAMES=NO;
7       DBDSOPTS='RENAME=(F1=Subsidiary F2=Stores F3=Sales F4=Inventory)';
8       RANGE=small_range;
9   RUN;

NOTE: WORK.SHOES data set was successfully created.
NOTE: The data set WORK.SHOES has 7 observations and 4 variables.
NOTE: PROCEDURE IMPORT used (Total process time):
      real time          0.17 seconds
      cpu time           0.06 seconds

10  PROC DATASETS LIBRARY=work NOLIST;
11     NOTE: Writing HTML Body file: sashtml.htm
12     MODIFY shoes;
13     LABEL Subsidiary = "Subsidiary"
14         Stores = "Stores"
15         Sales = "Sales"
16         Inventory = "Inventory";
17  QUIT;

NOTE: MODIFY was successful for WORK.SHOES.DATA.
NOTE: PROCEDURE DATASETS used (Total process time):
      real time          0.25 seconds
      cpu time           0.15 seconds
```

Figure 3.3: The SAS Dataset Created by the Code Above.

Example 3.4 PROC IMPORT Using the DBMS=EXCELCS Option

This example is similar to Example 3.2, but the code was executed on a Windows 64-bit configuration. The 64-bit operating system requires the use of the PC Files Server to execute any PROC IMPORT code where DBMS=EXCELCS. The SAS code for Part 1 reads the full Excel worksheet. The difference in the code is the use of the DBMS=EXCELCS option. Note that in most cases the “named-constants” are used as part of the
syntax of the \texttt{RANGE=} option; the “named-constants” are not required when a range-name is used with the \texttt{RANGE=} statement.

**Example 3.4 – Part 1**

The following SAS code reads a full worksheet from an Excel file on a 64-bit computer; the \texttt{DBMS=EXCELCS} option uses the SAS PC Files Server to access and read the input Excel 32-bit workbook.

```
PROC IMPORT
  DATAFILE='c:\My_Files\Shoes.xlsb'
  DBMS=EXCELCS
  OUT=shoes
  REPLACE;
  RANGE='SHOES$'n;
RUN;
```

**Example 3.4 – Part 2**

The following segment of SAS code, while syntactically correct, reads the first row of data as variable names and produces unpredictable results because \texttt{GETNAMES=} is not supported when \texttt{DBMS=EXCELCS}. This code is intended to read three rows of data from the input Excel file. However, the first row is interpreted as SAS variable names.

\textbf{NOTE:} The \texttt{RANGE=} value includes Excel cell references, which may not produce your desired output because the \texttt{GETNAMES=} statement is not supported when using the \texttt{DBMS=EXCELCS} option. I suggest that you use the \texttt{DBMS=XLSX} option instead, as shown in Example 3.5. This example shows what happens if you do not use the \texttt{DBMS=XLSX} statement.

```
/* this code does not work */
PROC IMPORT
  DATAFILE='c:\My_Files\Shoes.xlsb'
  DBMS=EXCELCS
  OUT=shoes
  REPLACE;
  RANGE='shoes$C2:F4'n;
RUN;
```

Figure 3.4 shows the output SAS dataset generated by the \texttt{PROC IMPORT} code from above. The intended result was to read three data rows into the SAS dataset. However, the first row was read and translated into variable names.

**Figure 3.4: The SAS Dataset Created by the Code Above.**
Example 3.5 PROC IMPORT Using the DBMS=XLS or XLSX to Select Columns

When using the DBMS=XLS option of PROC IMPORT with the ENDCOL and STARTCOL statements, the output SAS dataset is restricted to only the columns requested. This works like a KEEP statement, except the columns have to be contiguous. The input file is the SASHELP.SHOES dataset as exported to an Excel file. This example imports columns 2, 3, and 4 (Product, Subsidiary, and Number of Stores).

NOTE: There is a comment in the SAS log about a name change for the variable named “Number of Stores” because this text value has spaces embedded in the value. The value shown in Figure 3.5a for column 3 (Number of Stores) is the label applied to the variable named “Number_of_Stores”. Also, ENDCOL= was placed before STARTCOL= to show the statement order is not important. The output SAS dataset has data from three rows and five columns of the input Excel worksheet.

```
PROC IMPORT
    DATAFILE='c:\My_Excel_Files\Shoes.xls'
    DBMS=XLS
    OUT=shoes
    REPLACE;
    ENDCOL=4;
    STARTCOL=2;
RUN;
```

The system output log for Example 3.5 shows the name change of the variable “Number of Stores.” The log also verifies that only three columns were output to the SAS dataset from Excel.

```
1   PROC IMPORT
2       DATAFILE='c:\My_Excel_Files\Shoes.xls'
3       DBMS=XLS
4       OUT=shoes
5       REPLACE;
6       ENDCOL=4;
7       STARTCOL=2;
8   RUN;
```

NOTE: Variable Name Change. Number of Stores -> Number_of_Stores
NOTE: The import data set has 395 observations and 3 variables.
NOTE: WORK.SHOES data set was successfully created.
NOTE: PROCEDURE IMPORT used (Total process time):
real time           0.03 seconds
cpu time            0.04 seconds

SAS output dataset:
In Figure 3.5a, the SAS dataset label shown for the variable Number_of_Stores has two spaces; however, the actual variable name does not have any spaces embedded.
Example 3.6 PROC IMPORT Using the DBMS=XLS or XLSX to Select Rows

This example uses the PROC IMPORT option pairs STARTROW= / ENDROW= and STARTCOL= / ENDCOL= to show you how you can select a range of cells from an Excel worksheet without creating a named range in an Excel workbook. When the NAMEROW=, GETNAMES=, and RANGE= statements are added to the mix, you can pick names for your variable from inside the Excel file without needing a second pass over the dataset or the need to use PROC DATASETS. The text values with spaces embedded in the value have had an underscore added to replace the space in the variable name. Also, ENDROW= was placed before STARTROW= to show the statement order in not important. The output SAS dataset has data from three columns and five rows of the input Excel worksheet.

```
PROC IMPORT
  DATAFILE='c:\My_Files\Shoes.xls'
  DBMS=XLS
  OUT=shoes
  REPLACE;
  ENDCOL="4";  /* a quoted string is required */
  STARTCOL="2";  /* a quoted string is required */
  ENDROW=10;  /* numeric value is required */
  STARTROW=6;  /* numeric value is required */
  NAMEROW=1;
  GETNAMES=NO;
RUN;
```
Output Log of Code Above

```sas
1 2 3      PROC IMPORT 4          DATAFILE='c:\My_Files\Shoes.xls' 5        DBMS=XLS 6        OUT=shoes 7        REPLACE; 8        ENDCOL="4";  /* a quoted string is required */ 9        STARTCOL="2";  /* a quoted string is required */ 10       ENDRON=10;  /* numeric value is required */ 11       STARTRON=6;  /* numeric value is required */ 12       NAMEROW=1; 13       GETNAMES=NO; 14       RUN;
NOTE:    Variable Name Change.  Number of Stores -> Number_of_Stores
NOTE: The import data set has 5 observations and 3 variables.
NOTE: WORK.SHOES data set was successfully created.
NOTE: PROCEDURE IMPORT used (Total process time):  
      real time           0.06 seconds
      cpu time            0.01 seconds
```

Figure 3.5b: Using PROC IMPORT to Select Rows and Headers from an Excel Worksheet.

Example 3.7 PROC IMPORT Using the DBMS=XLS or XLSX to Select Excel Ranges

This example was executed on a computer running 64-bit Windows 8.1 Professional on 64-bit hardware with SAS 9.4 and 32-bit Excel 2013 installed. The DBMS option XLSX provides an alternative method to reading a small group of cells from an Excel spreadsheet. However, this method does not always provide reliable variable names when GETNAMES=YES. GETNAMES=YES looks for variable names in the first row of input cells. Here, GETNAMES=NO is used to turn off the search for variable names in the Excel file. The RANGE='shoes$C2:F4'n command selects only 12 cells from the Excel file.

```sas
PROC IMPORT  
            DATAFILE='c:\My_Files\Shoes.xlsx'  
            DBMS=XLSX  
            OUT=shoes  
            REPLACE;  
            GETNAMES=NO;  
            RANGE='shoes$C2:F4'n;  
RUN;
```
3.7 Conclusion

I have shown several methods of reading data and variable names from Excel workbooks. But, there are far too many other combinations of options available for me to present an exhaustive list. This chapter showed features of PROC IMPORT. Some of the important items to take away from this chapter are that the Microsoft Excel JET and ACE engines have limitations. These limitations will occasionally affect the amount of data you can extract from your Excel files. There may be times when you are required to fall back to the tried-and-true delimited file formats to transfer your data to and from Excel. I suggest that you refer to SAS/ACCESS Interface to PC Files: Reference for the version of SAS that you have installed. These documents have SAS version-specific descriptions of the syntax and features available for the SAS Import Wizard and PROC IMPORT.

Index

A

ABSOLUTE_COLUMN_WIDTH option 130, 134–138
Access Connectivity Engine (ACE) 1, 159
accessing
  data from SAS Explorer window/toolbar 9–18
  data stored in workbooks 241–245
  Excel data from SAS Explorer window 9–18
  Excel files with LIBNAME 48–49
  Excel files with SQL procedure 157–166
  Excel from SAS Enterprise Guide 68
  Excel with OLE DB or ODBC application program interfaces (APIs) 149–156
  Excel with OLE DB/ODBC APIs 149–156
  SAS Export/Import wizards 15–16
activating UserForm 248–258
Add-In for Microsoft Office
  See SAS Add-In for Microsoft Office
applying Excel “AUTOFILTER” to selected output columns 136–137
ASCII_DOTS option 131
assigning
  LIBNAME statement 160–161
  Libref 52, 151–152
ATTRIB command 48
AUTOFILTER option
  about 131, 136–137
  applying to selected output columns 136–137
AUTOFILTER_TABLE option 131
AUTOFIT_HEIGHT option 131
automating processing 197–201
AUTO_SUBTOTALS option 131

B

bar charts
  building using SAS Enterprise Guide 99–100
  exporting as *.srx files 100–102
BLACKANDWHITE option 131
BLANK_SHEET option 132
building
  bar charts using SAS Enterprise Guide 99–100
  Excel macros 189–209
  Excel user form interface 233–248
  VBS scripts 220–223
BY statement 143
BYGROUP setting 143–144
BYLINES option 112

C

calculating variables within SQL code 165
cell ranges
  defining 171, 172–173
  reading from Excel workbooks 163–165
  reading with Import Data option 80–83
CENTER_HORIZONTAL option 131
CENTER_VERTICAL option 131
changes, saving 255
character fields
  about 2
  saving leading zeros in 123–124
CheckBox 234
client-server environment (Dynamic Data Exchange (DDE)) 168–170
Close_Excel macro 171, 174–175
closing Excel workbooks 171, 174–175
COLUMN_REPEATE option 130
columns
  adjusting width with tagset template options 134–135
  hiding 135–136
  selecting 42–43
ComboBox 233
CommandButton 234
commands, issuing to Excel 171, 172
  See also specific commands
comparing Excel files 161–162
copy-and-paste techniques
  about 4
  converting Excel tables to text 9
  converting text data to Excel column data fields 5–7
Component Object Model (COM) software package 91
CONNECTION option, LIBNAME statement 50–53
CONNECTION TO statement 158–159
CONTENTS option 132, 139
CONTENTS procedure
  about 57, 165–166
  examining Excel workbook with 60–62
  verifying Excel to OLE DB connection with 154–156
CONTENTS_WORKBOOK option 132
ControlSource 234
CONVERT_PERCENTAGES option 131
“Copy Contents to Clipboard” option (SAS Explorer window) 11–12
copying data to SAS Enhanced Editor window 7–8
highlighting cut/copy and then paste 5
saving multiple lines of text 8
copying data to SAS Enhanced Editor window 7–8
source programs 253–254
CREATE TABLE command 166
creating
Excel files 161–162
Excel workbooks 231–258
naming conventions 214–215
output files 108–148, 202–203
“ready-to-print” spreadsheet 137–138
table of contents in Excel workbook 138–139
worksheets 143–145
CSS_TABLE option 125
CSV option
about 109, 111–112
changing delimiters when outputting data with 120–123
examples 113–123
file default output differences 113–115
overview of examples 113
saving leading zeros in character fields 123–124
title and footnotes output differences 115–118
writing currency values as unformatted numbers 118–120
CSVALL option
about 109, 111–112
examples 113–123
file default output differences 113–115
overview of examples 113
saving leading zeros in character fields 123–124
title and footnotes output differences 115–118
writing currency values as unformatted numbers 118–120
currency values, writing as unformatted numbers 118–120
currency values, writing as unformatted numbers 118–120
CURRENCY_AS_NUMBER option 112
CURRENCY_FORMAT option 131
CURRENCY_SYMBOL option 112, 131
D
data
access methods for Excel files supported by IMPORT procedure 33–34
accessing from SAS Explorer window(toolbar 9–18
copying to Excel files via HTML files with “View in Excel” option 13–14
copying to SAS Enhanced Editor window 7–8
exporting to Excel 4/5 format files 23–24
exporting to Excel files with no column headers 28
exporting to network Windows computers 28
loading to UserForms 251–253
options for in SAS Add-In for Microsoft Office 96–99
ranges of in Excel 4
reading from Excel to JMP 88–89
sharing with Excel using JMP 85–89
types of 2, 50–51
writing from JMP to Excel 89
writing to external files and Excel workbooks with EXPORT procedure 19–29
Data Base Management System (DBMS) 31
DATAFILE statement 17
dataset options
processing date and time values with 62–63
processing variable type conversions with 63–64
datasets, opening 94–96
date values, processing with dataset options 62–63
DBCREATE_TABLE_OPTS option 53
DBDSOPTS= option 39
DBENCODING option 53
DBFORCE option 53
DBGEN_NAME option 53, 163–165
DBLABEL option 53
DBMAX_TEXT option 53
DBMS mode 21–22
DBMS=DLM option
EXPORT procedure 24–25
IMPORT procedure 35–37
DBMS=EXCEL option
EXPORT procedure 25–27
IMPORT procedure 37–40
DBMS=EXCEL4 option, IMPORT procedure 35
DBMS=EXCEL5 option, IMPORT procedure 35
DBMS=EXCELCS option
EXPORT procedure 27–28
IMPORT procedure 40–41
DBMS=XLS option, IMPORT procedure 42–45
DBMS=XLSX option, IMPORT procedure 42–45
DBSASLABEL option 53
DBSASTYPE option 53, 63–64
DBTYPE option 53
DDE
See Dynamic Data Exchange (DDE)
DECIMAL_SEPARATOR option 112, 131
DEFAULT_COLUMN_WIDTH option 130
defining
cell ranges 171, 172–173
physical file locations 51–52
DELIMITER option 112
delimiters, changing 120–123
directories, structure of 255–256
DOC option 112, 125, 132
DPI option 131
DRAFTQUALITY option 131
DROP option  54
Dynamic Data Exchange (DDE)
   about  167, 213
   client-server environment  168–170
   examples  177–187
   Hello World project  177–181
   list of examples  177
   macros for  171–177
   purpose of  167–168
   reading and writing to Excel workbooks with  167–187
   syntax of  168–170
   writing “Hello World” to Excel files  182–184
   writing SAS datasets to Excel files  184–187

E
EMBEDDED_FOOTNOTES option  130
EMBEDDED_TITLES option  130
EMBED_TITLES_ONCE option  130, 146
Enhanced Editor window, copying data to  7–8
Enterprise Guide
   See SAS Enterprise Guide
examples
   CSV option  113–123
   CSVALL option  113–123
   Dynamic Data Exchange (DDE)  177–187
   EXCELXP option  133–146
   JMP  87–89
   LIBNAME statement  56–65
   MSOFFICE2K option  126–128
   OLE DB/ODBC APIs  151–156
   SAS Add-In for Microsoft Office  94–105
   SAS Enterprise Guide  69–83
   SQL procedure  160–66
Excel
   See also Excel files
   See also workbooks (Excel)
   See also worksheets (Excel)
   accessing from SAS Enterprise Guide  68
   accessing with OLE DB or ODBC application
      program interfaces (APIs)  149–156
   building control macros for reports  223–229
   building graphs with macros  207–209
   building macros  189–209
   building user form interface  233–248
   converting tables to text  9
   data ranges  4
   data types  2
   executing UserForm  245–248
   guidelines for building user form interface  233–248
   importing *.srx files into  102–105
   issuing commands to  171, 172
   LIBNAME assignments to access Excel using SQL
      procedure  160–161
   purpose of  2
   reading data to JMP from  88–89
   selecting ranges  44–45
   sharing data with using JMP  85–89
   sharing methods between JMP and  86–87
   starting  171–172
   verifying to OLE DB connection with CONTENTS
      procedure  154–156
   workbook formatting groups  3–4
   workbook limitations  2–3
   writing data from JMP to  89
Excel files
   accessing with LIBNAME  48–49
   accessing with SQL procedure  157–166
   comparing  161–162
   copying data to via HTML files with “View in
      Excel” option  13–14
   creating  161–162
   data access methods for files supported by
      EXPORT procedure  21–22
   data access methods for files supported by
      IMPORT procedure  33–34
   exporting data to  23–24, 28
   exporting data to with no column headers  28
   processing with LIBNAME statement  47–65
   processing with pass-through facilities using SQL
      procedure  162–163
   reading  161–162
   writing “Hello World” to  182–184
   writing SAS datasets to  184–187
EXCEL ODS destination, for writing workbooks  147–148
EXCEL option  109
Excel Workbook Tool  212, 232
Excel-readable files  4
Excel-specific dataset options  53–54
EXCELXP option  109, 189–190
   adjusting column width with tagset template
      options  134–135
   applying Excel “AUTOFILTER” to selected output
      columns  136–137
   building worksheets with titles  146
   creating “ready-to-print” spreadsheet  137–138
   creating table of contents in Excel workbook  138–139
   examples  133–146
   generating XML output files with no options  133–134
   hiding columns  135–136
   naming Excel worksheets  140
   overview of examples  132–133
   placing labels in names of Excel worksheets  142–143
   splitting reports onto multiple Excel worksheets
      141
   syntax of  128–130
tagset options 130–132
executing
  Excel UserForm 245–248
  VBS/VBA macros 214–215
  Visual Basic Script 219–220
exiting programs 255
Explorer window
  See SAS Explorer window
Export method
  outputting graphs or reports with 75–77
  using with SAS Enterprise Guide 69–70
EXPORT procedure
  about 9, 213
  data access methods for Excel files supported by 21–22
  DBMS=DLM option 24–25
  DBMS=EXCEL option 25–27
  DBMS=EXCELCS option 27–28
  examples 23–28
  exporting data to Excel 4/5 format files 23–24
  exporting data to Excel files with no column headers 28
  exporting data to network Windows computers 28
  list of examples 23, 150, 177
  overview of examples 22–23
  purpose of 20
  syntax of 20–21
  writing SAS data to external files and Excel workbooks with 19–29
Export wizard
  accessing 15–16
  selecting from SAS Explorer window “Export” menu 11
  using in 32/64-bit mixed environment 17–18
  exporting bar charts as *.srx files 100–102
  external files
    reading into SAS with IMPORT procedure 31–45
    writing SAS data to with EXPORT procedure 19–29
F
file format groups 3
File option (SAS Toolbar) 15–16
file output, opening 192–194
FILENAME statement 50–51, 177–179
FILEREF 173
files
  See Excel files
  See external files
  See HTML files
  See output files
FITTOPAGE option 131
formulas 2
FORMULAS option 131
FROZEN_HEADERS option 130, 137–138
FROZEN_ROWHEADERS option 130
G
GETNAMES= statement 38, 39, 41, 43–44
graphs, outputting 71–77
GRIDLINES option 131
GUESSINGROWS option 57
H
hardware configuration 49
HEADER option, LIBNAME statement 49, 50, 56–57
HEADER_DATA_ASSOCIATIONS option 125
HEADER_DOTS option 125
Hello World project 177–181, 189–190
HIDDEN_COLUMNS option 131, 135–136
hiding columns 135–136
highlighting cut/copy and then paste 5
HTML files
  copying data to Excel files via with “View in Excel” option 13–14
  generating with no options 126–127
  generating with Summary_Vars option 127–128
  writing 124–125, 126–128, 133–146
  HTML option 109
I
IDE
  See Integrated Development Environment (IDE)
Import Data option (SAS Toolbar) 16–17, 80–83
Import option, reading Excel workbooks with 77–80
IMPORT procedure
  about 9, 31–32
  data access methods for Excel files supported by 33–34
  DBMS=DLM option 35–37
  DBMS=EXCEL option 37–40
  DBMS=EXCEL4 option 35
  DBMS=EXCEL5 option 35
  DBMS=EXCELCS option 40–41
  DBMS=XLS option 42–45
  DBMS=XLSX option 42–45
  examples 35–45
  list of examples 34–35
  overview of examples 34
  purpose of 32
  reading external data files and Excel workbooks into SAS with 31–45
    syntax of 32
Import wizard
  accessing 15–16
  using in 32/64-bit mixed environment 17–18
  importing *.srx files into Excel 102–105
IN option 54
INDEX option 132
INFILE statement  7
initializing
  active UserForm values  250–251
  passive UserForm values  249–250
  UserForm  248–249
Insert group (SAS Add-In for Microsoft Office)  93
INSERT_SQL option  53
Integrated Development Environment (IDE)
  about  233, 235–236
  accessing data stored in workbooks  241–245
  building sample menus  237–238
  executing Excel UserForm  245–248
  linking menu and data  238–239
  setting up control variables  241–245
  storing control information in Excel workbook

  worksheets  240–241
  Toolbox menu  236–237
issuing commands to Excel  171, 172

J
JMP
  about  85
  examples  87–89
  list of examples  87
  purpose of  85–86
  reading data from Excel to  88–89
  setting preferences  87–88
  sharing data with Excel using  85–89
  sharing methods between Excel and  86–87
  writing data to Excel from  89
Joint Engine Technology (JET) database engine  1, 159

K
KEEP option  54

L
Label  233
labels, placing in Excel worksheet names  142–143
leading zeros, saving in character fields  123–124
LIBNAME statement
  about  16, 47–48
  assigning to access Excel using SQL procedure

  160–161
  building OLE-DB connection with prompt mode

  152–153
  CONNECTION option  50–53
  examining Excel workbooks with CONTENTS

  procedure  60–62
examples  56–65
Excel-specific dataset options  53–54
Excel-specific features of  48–49
HEADER option  49, 50, 56–57
LINUX option  54
MIXED option  49, 50–51, 57
overview of examples  55
PATH option  49, 51–52, 58, 162–163
processing date and time values with dataset

  options  62–63
processing Excel files with  47–65
processing on 64-bit operating system  64–65
processing variable type conversions with dataset

  options  63–64
PROMPT option  50, 52
purpose of  48
64-bit Windows connection option  54
syntax of  49–50
UNIX option  54
uses for  49
using named literals with  59–60
VERSION option  50, 52, 58–59
Libref
  assigning  52, 151–152
  assigning to Excel worksheets with OLE-DB dialog

  box  151–152
LINUX option  54
loading data to UserForms  251–253
LRECL= option  180–181

M
Macro Library Tool  212, 232
macros
  building Excel graphs with  207–209
  building for Excel  189–209
  controlling with Microsoft Windows scripts  211–

  229
  for Dynamic Data Exchange (DDE)  171–177
MERGE_TITLES_FOOTNOTES option  130
Microsoft Excel
  See Excel
Microsoft Windows scripts, controlling with macros
  211–229
MINIMIZE_STYLE option  132
MISSING_ALIGN option  130
MIXED option, LIBNAME statement  49, 50–51, 57
MSOFFICE2K option
  about 109
  examples  126–128
  generating HTML files with no options  126–127
  generating HTML files with Summary_Vars option

  127–128
overview of examples  126
syntax of output processes  124–125
tagset template options  125

N
Name  234
named literals, using with LIBNAME statement  59–60
NAMEROW= statement  43–44
naming conventions, creating  214–215
naming Excel worksheets 140
network Windows computers, exporting data to 28
NEWFILE= option 27, 28
NONE option, SHEET_INTERVAL option 143
NOTAB option 180–181
NOTES option 112
numeric values 2
NUMERIC_TEST_FORMAT option 131

O

ODS
See Output Delivery System (ODS)
OLE DB LIBNAME, syntax of 150
OLE DB/ODBC APIs
accessing Excel with 149–156
assigning Libref to Excel worksheets with OLE-DB dialog box 151–152
building OLE-DB connection with LIBNAME prompt mode 152–153
concept of processes 149
examples 151–156
list of examples 150
opening Excel workbooks with OLE-DB init_string 154
setting up connections 150
verifying Excel to OLE DB connection with CONTENTS procedure 154–156
OLE-DB dialog box, assigning Libref to Excel worksheets with 151–152
OPEN option, reading Excel workbooks with 77–80
Open_cmd macro 171, 172
opening datasets 94–96
Excel workbooks with OLE-DB init_string 154
file output 192–194
report datasets (*.srx) using SAS Add-In for Microsoft Office 99–105

options
See specific options
ORIENTATION option 130, 137–138
output
graphs 71–77
processing 191–192
processing reports 258
reports 71–75, 75–77
sorting 165
verifying batch files 254–255
Output Delivery System (ODS)
about 108
creating output files with 108–148
ODS tagset compared with ODS destination 111
purpose of 108–109
SAS Tagset templates 109–110
syntax of CSV and CSVALL output processes 111
output files
creating 108–148, 202–203
creating with ODS (Output Delivery System) 108–148
processing 203–206
Out_range macro 171, 172–173

P

PAGE option, SHEET_INTERVAL option 143
PAGE_BREAK option 125, 131
PAGE_ORDER_ACROSS option 131
PAGES_FITHEIGHT option 131
PAGES_FITWIDTH option 131
“PASS-THROUGH” processing 160
PATH option, LIBNAME statement 49, 51–52, 58, 162–163
PCFILES special query 159, 165–166
PERCENTAGE_AS_NUMBER option 112
PERCENTAGE_FONT_SIZE option 125
Personal Workbook Tool 211, 232
physical file locations, defining 51–52
placing labels in Excel worksheet names 142–143
preferences (JMP) 87–88
PREPEND_EQUALS option 112
PRINT procedure 5, 115, 143, 191, 196, 202, 208
PRINT_FOOTER option 130
PRINT_FOOTER_MARGIN option 130
PRINT_HEADER option 130
PRINT_HEADER_MARGIN option 130
PROC option, SHEET_INTERVAL option 143, 144–145
processing
automating 197–201
date and time values with dataset options 62–63
Excel files with LIBNAME statement 47–65
Excel files with pass-through facilities using SQL procedure 162–163
output 191–192
output files 203–206
output reports 258
on 64-bit operating system 64–65
variable type conversions with dataset options 63–64
PROC_TITLES option 112
production directories, copying source programs to working directories from 253–254
programs, exiting 255
PROMPT option, LIBNAME statement 50, 52
proof-of-concept program 190
Properties Window 234
PropertySheet 234
PUTNAMES=NO option 28, 36
INDEX 267

Q

QUOTE_BY_TYPE option 112
QUOTED_COLUMNS option 112

R

RANGE= statement 37, 41, 43–44
READBUFF option 53
reading
    cell ranges from Excel workbooks 163–165
    cell ranges with Import Data option 80–83
    data from Excel to JMP 88–89
    Excel files 161–162
    to Excel workbooks with Dynamic Data Exchange (DDE) 167–187
    Excel workbooks with Open or Import options 77–80
    external data files and Excel workbooks into SAS with IMPORT procedure 31–45
    pre-defined cell ranges from Excel workbooks 163–165
    subsets of records from Excel workbooks with SQL procedure 162
    variable names 50
"ready-to-Print" spreadsheet, creating 137–138
reformatting Excel workbooks 194–197
RENAME option 39, 54
report datasets (*.srx), opening using SAS Add-In for Microsoft Office 99–105
reports
    outputting 71–75, 75–77
    splitting onto multiple Excel worksheets 141
ROWCOLHEADINGS option 131
ROW_HEIGHT_FUDGE option 131
ROW_HEIGHTS option 131
ROW_REPEAT option 131
rows, selecting 43–44

S

Safe Mode 233
SAS Add-In for Microsoft Office
    about 91
    data options for 96–99
    examples 94–105
    list of examples 94
    methods of sharing data using 92–93
    opening datasets using 94–96
    opening report datasets (*.srx) using 99–105
    purpose of 91
SAS Enhanced Editor window, copying data to 7–8
SAS Enterprise Guide
    about 67–68
    accessing Excel from 68
    building bar charts using 99–100
    examples 69–83
    exporting bar charts as *.srx files from using 100–102
    list of examples 68–69
    methods 67–83
    overview of examples 68
    reading cell ranges with Import Data option 80–83
    reading Excel workbooks with Open or Import options 77–80
    “Send To” method 71–75
    using Export method with 69–77
SAS Explorer window
    about 10–11
    accessing Excel data from 9–18
    “Copy Contents to Clipboard” option 11–12
    “Save as Html” option 12–13
    selecting Export wizard from “Export” menu 11
SAS Toolbar
    File option 15–16
    “Import Data” option 16–17
SAS_2_Excel macro 171, 175–176, 184–187
SASDATEFMT option 53
“Save as Html” option (SAS Explorer window) 12–13
Save_Excel macro 171, 174
saving
    changes 255
    Excel workbook contents 171, 174
    leading zeros in character fields 123–124
    lines of text 8
SCALE option 131
SELECT statement, ordering of clauses in 158–159
selecting
    columns 42–43
    data types 50–51
    Excel ranges 44–45
Selection group (SAS Add-In for Microsoft Office) 93
“Send To” method 71–75
SET statement 59
setup, workstation options 215–217
sharing
    data using SAS Add-In for Microsoft Office 92–93
    data with Excel using JMP 85–89
SHEET_INTERVAL option
    about 131
    creating worksheets with BYGROUP setting 143–144
    creating worksheets with PROC setting 144–145
SHEET_LABEL option 132, 142–143
SHEET_NAME option 132, 140, 141
simple file formats 4
64-bit operating system
    processing on 64–65
    using Export/Import wizards in 17–18
    Windows connection option 54
SKIP_SPACE option 132
software configuration 49
sorting output 165
source programs, copying 253–254
splitting reports onto multiple Excel worksheets 141

SQL procedure
about 57, 157
accessing Excel files with 157–166
calculating variables within SQL code 165
comparing Excel files 161–162
creating Excel files 161–162
examples 160–166
LIBNAME assignments to access Excel using 160–161
list of examples 160
“PASS-THROUGH” processing 160
“PCFILES::” special query 165–166
processing Excel files with pass-through facilities 162–163
purpose of 158
reading Excel files 161–162
reading pre-defined cell ranges from Excel workbooks 163–165
reading subsets of records from Excel workbooks with 162
sorting output 165
syntax of 158–159

*.srx files
exporting bar charts as 100–102
importing into Excel 102–105
Start_Excel macro 171–172
starting Excel 171–172

statements
See specific statements
storing
VBS/VBA macros 214–215, 217–219
VBS/VBA scripts 217–219
structure, of directories 255–256
SUMMARY option 125, 130, 202
SUMMARY_AS_CAPTION option 125, 130
SUMMARY_BYVALS option 125, 130
SUMMARY_BYVARS option 125, 130
SUMMARY_PREFIX option 125, 130
Summary_Vars option, generating HTML files with 127–128
SUPPRESS_BYLINES option 132
syntax
of CSV and CSVALL output processes 111
of Dynamic Data Exchange (DDE) 168–170
of EXCELXP option 128–130
of EXPORT procedure 20–21
of IMPORT procedure 32
of LIBNAME statement 49–50
of MSOFFICE2K output processes 124–125
of OLE DB LIBNAME 150
of SQL procedure 158–159

TABLE_HEADERS option 112
tables, converting to text in Excel 9
tagset templates
about 109–110
EXCELXP option 130–132
MSOFFICE2K option 125
TEMPLATE procedure 109–110, 113
text
converting Excel tables to 9
converting to Excel column data fields 5–7
saving multiple lines of 8

TextBox 233
32-bit operating system, using Export/Import wizards in 17–18

THOUSANDS_SEPARATOR option 112, 131
time values, processing with dataset options 62–63
TITLE_FOOTNOTE_WIDTH option 130
titles, building worksheets with 146
TITLES option 112
toolbar processing method 10–11
Tools group (SAS Add-In for Microsoft Office) 93

U
unformatted numbers, writing currency values as 118–120
UNIX option 54
user parameters, writing to files in working directories 253

UserForm
about 233
activating 248–258
initializing 248–249
loading data to 251–253

V
variables
calculating within SQL code 165
processing type conversions of with dataset options 63–64
reading names for 50
writing to Excel workbooks 171, 175–176
VBA routines 248–258
VBS/VBA macros
executing 214–215
guidelines for building and using 213–229
preparing 258
storing 214–215, 217–219
VBS/VBA scripts
building 220–223
storing 217–219

verifying output batch files 254–255
VERSION option, LIBNAME statement 50, 52, 58–59
“View in Excel” option, copying data to Excel files via HTML with 13–14
Visual Basic Scripting (VBS) 211, 219–220

table of contents, creating in Excel workbook 138–139
TABLE option, SHEET_INTERVAL option 143

T
W
WHERE statement  5, 54, 115
WIDTH_FUDGE option  131
WIDTH_POINTS option  131
Workbook_Open macro  198–201, 202–203
workbooks (Excel)
closing  171, 174–175
creating table of contents in  138–139
creating that runs SAS programs  231–258
examining contents and structure of  165–166
examining with CONTENTS procedure  60–62
formatting groups  3–4
limitations of  2–3
opening with OLE-DB init_string  154
reading into SAS with IMPORT procedure  31–45
reading pre-defined cell ranges from  163–165
reading subsets of records from with SQL procedure  162
reading to with Dynamic Data Exchange (DDE)  167–187
reading with Open or Import options  77–80
reformatting  194–197
saving contents of  171, 174
writing  147–148
writing SAS data to with EXPORT procedure  19–29
writing to with Dynamic Data Exchange (DDE)  167–187
writing variables to  171, 175–176
working directories
copying source programs to production directories from  253–254
writing user parameters to files in  253
worksheets (Excel)
assigning Libref with OLE-DB dialog box  151–152
building with titles  146
creating with SHEET_INTERVAL=BYGROUP option  143–144
creating with SHEET_INTERVAL=PROC option  144–145
naming  140
placing labels in names of  142–143
splitting reports onto multiple  141
workstation options, setting up  215–217
WRAPTEXT option  130, 137–138
writing
currency values as unformatted numbers  118–120
data from JMP to Excel  89
to Excel workbooks with Dynamic Data Exchange (DDE)  167–187
“Hello World” to Excel files  182–184
HTML files  124–125, 126–128, 133–146
SAS datasets to Excel files  184–187
user parameters to files in working directories  253
variables to Excel workbooks  171, 175–176
worksheets  147–148
X
X command  179–180, 202, 219–220
XLSTART directory  193–194
XML output files, generating with no options  133–134
Z
ZOOM option  130, 137–138
About This Book

Purpose
I wrote this book to help SAS users of all skill levels find out how to move data between SAS and Microsoft Excel. My years of programming experience have helped me decode the mysteries of vendor-supplied system documentation. I wanted to gather that information together and present it in an easy-to-understand tutorial format with the prime emphasis on examples. I have also scattered in my observations on the world of programming in general and pieced together an array of examples that include both simple and complex task descriptions.

Is This Book for You?
Whatever your skill level, I hope you will find examples that will teach you something. In every class I teach or paper I present, I always ask if anyone learned anything. I want you to be able to find a place on your desk for this book, use it as you progress through the skills presented, and gain expertise to easily move your data.

Prerequisites
This book is designed for you to use without need for prerequisites. If you can open the SAS program and copy data using your mouse, then you can get started. I do not attempt to teach you how to write SAS programs or build an Excel spreadsheet, but I present methods to move data between the two data storage tools.

Scope of This Book
This book attempts to show you how to move data “BETWEEN” SAS and Excel. I have attempted to use as many differing techniques as I could within the limited space available. As I worked my way through the chapters, I created examples that progressively increased in power and complexity.

But, what I do not do is show you very much about how to use the data after it is moved or copied into either Excel or SAS. Within this book I have covered many ways that show you how to shuffle your data between SAS and Excel. I hope I have also opened ways to manipulate the worksheets after they have been written. I have tried to keep the data simple and only change the methods. In fact, nearly every example uses the same SAS dataset, as noted below.

About the Examples
Software Used to Develop the Book’s Content
Because SAS users are likely to be working with different SAS versions, I have included examples that use several versions of SAS software. Most of the examples use SAS 9.4. Some JMP examples and SAS Enterprise Guide examples are also shown. Examples of Excel screens also vary across several versions of Excel, from Excel 2003 to Excel 2013. The examples in the book cover the transition from the xls workbooks to the xlsx workbooks and the way SAS has adapted to those Excel changes.
Example Code and Data

The primary dataset used for examples in this book is the SASHELP.SHOES SAS dataset; it is used as an exported file to Excel and then as input from Excel. The SASHELP.SHOES dataset is shipped with every version of SAS and is therefore convenient for all users.

You can access the example code and data for this book by accessing my author page at http://support.sas.com/publishing/authors. Select the name of the author, look for the cover thumbnail of this book, and select Example Code and Data to display the SAS programs that are included in this book.

For an alphabetical listing of all books for which example code and data is available, see http://support.sas.com/bookcode. Select a title to display the book’s example code.

If you are unable to access the code through the website, email saspress@sas.com.

Additional Help

Although this book illustrates many analyses regularly performed in businesses across industries, questions specific to your aims and issues may arise. To fully support you, SAS Institute and SAS Press offer you the following help resources:

- For questions about topics covered in this book, contact the author through SAS Press:
  - Send questions by email to saspress@sas.com; include the book title in your correspondence.
  - Submit feedback on the author’s page at http://support.sas.com/author_feedback.

- For questions about topics in or beyond the scope of this book, post queries to the relevant SAS Support Communities at https://communities.sas.com/welcome.

- SAS Institute maintains a comprehensive website with up-to-date information. One page that is particularly useful to both the novice and seasoned SAS user is the SAS Knowledge Base. Search for relevant notes in the “Samples and SAS Notes” section of the Knowledge Base at http://support.sas.com/resources.

- Registered SAS users or their organizations can access SAS Customer Support at http://support.sas.com. Here you can pose specific questions to SAS Customer Support. Under Support, click Submit a Problem. You will need to provide an email address to which replies can be sent, identify your organization, and provide a customer site number or license information. This information can be found in your SAS logs.

Keep in Touch

We look forward to hearing from you. We invite questions, comments, and concerns. If you want to contact us about a specific book, please include the book title in your correspondence.

Contact the Author through SAS Press
- By email: saspress@sas.com
- Via the web: http://support.sas.com/author_feedback

Purchase SAS Books
For a complete list of books available through SAS, visit sas.com/store/books.

- Phone: 1-800-727-0025
- Email: sasbook@sas.com
Subscribe to the SAS Training and Book Report
Receive up-to-date information about SAS training, certification, and publications via email by subscribing to the SAS Training & Book Report monthly eNewsletter. Read the archives and subscribe today at http://support.sas.com/community/newsletters/training!

Publish with SAS
SAS is recruiting authors! Are you interested in writing a book? Visit http://support.sas.com/saspress for more information.
About The Author

William E. Benjamin, Jr., owns Owl Computer Consultancy, LLC, and works as a consultant, trainer, and author. William has been a SAS user for over 30 years and a consultant since 2007. He received an MBA from Western International University and a BS in computer science from Arizona State University. He has written and presented papers for SAS Global Forum, as well as many regional and local SAS users groups.

Learn more about this author by visiting his author page at http://support.sas.com/publishing/authors/benjamin.html. There you can download free book excerpts, access example code and data, read the latest reviews, get updates, and more.
Gain Greater Insight into Your SAS® Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.

support.sas.com/bookstore for additional books and resources.