

Importing EXCEL® Data in Different SAS® Maintenance Release Version

Huei-Ling Chen, Chao-Min Hoe, Merck & Co., Inc.

ABSTRACT

It was noted that using PROC IMPORT procedure to convert the same Excel data file to SAS dataset, the outputs could be inconsistent from different computers. The objective of this manuscript is to investigate the inconsistencies and to provide explanations and solutions. Several Excel data files were tested. It was observed that different outcomes were due to the SAS version or the maintenance release on different PCs.

KEYWORDS

PROC IMPORT, DBMS XLSX EXCEL, GETNAMES OPTION, MIXED OPTION, SYSVLONG, MAINTENANCE RELEASE VERSION

INTRODUCTION

BACKGROUND

Laboratory experiments are conducted on animals to determine the potential cardiovascular and respiratory effects after the administration of pharmaceutical compounds in development. Since these standard procedures are performed routinely, SAS macros were generated to conduct the statistical analysis and produce graphs and summary tables.

In a study team with multiple users, the computer for each team member might have different SAS versions or maintenance release. Different SAS versions could generate different statistical outputs, even the same SAS codes were used.

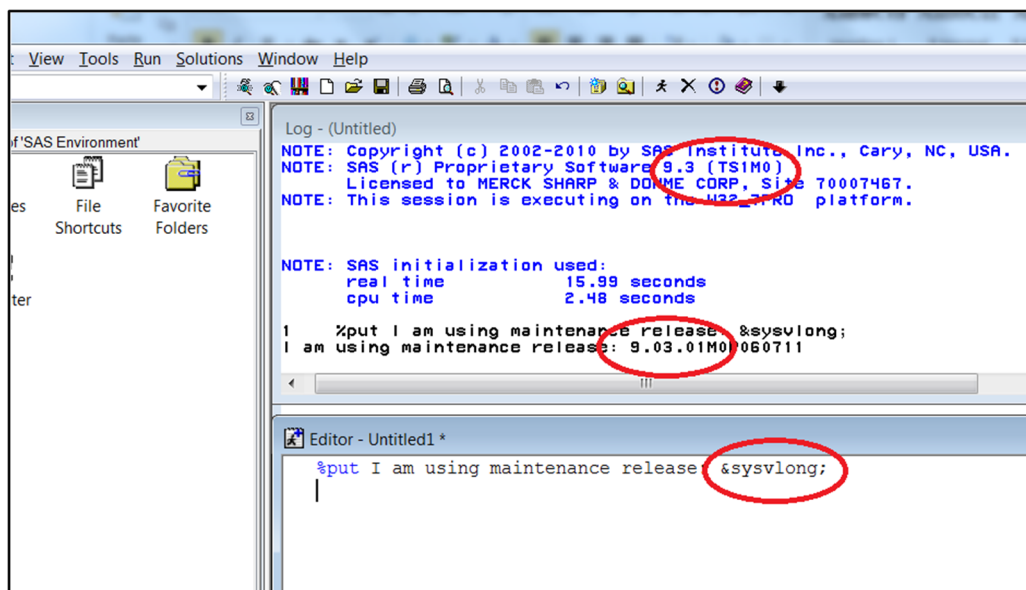
HOW TO FIND OUT THE SAS MAINTENANCE RELEASE VERSION

Method 1: Open SAS. The top 4 rows in the log window indicate the maintenance release version.

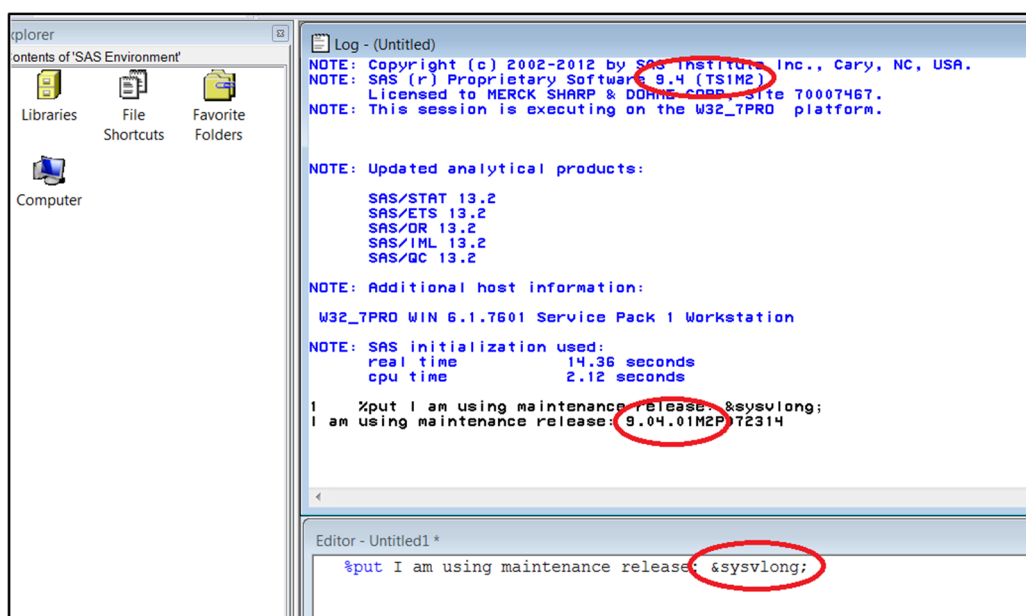
Method 2: SAS automatic macro variable SYSVLONG provides the release number and maintenance level of SAS software.

```
%put I am using maintenance release: &sysvlong;
```

The automatic macro variable SYSVLONG is utilized to direct SAS code using the correct procedure per different SAS version or maintenance release.



Display 1: SAS 9.3 TS1M0



Display 2: SAS 9.4 TS1M2

ISSUE 1: INCONSISTENT VARIABLE NAMES IN XLSX DRIVER

The IMPORT procedure reads in external data and writes it to a SAS dataset. The external data could be EXCEL data, ACCESS data, STATA data ... etc. The DBMS option specifies the type of data to be imported. Different results produced from DBMS=EXCEL and DBMS=XLSX are discussed below.

The IMPORT procedure generates SAS variable names from the data value of the first row. If data value of the first row is not a valid SAS name, then the IMPORT procedure using GETNAMES=NO option to generate the default SAS variable name.

DBMS XLSX driver creates the variable names such as letter 'A' for the first column, 'B' for the second column, 'C' for the third column... etc.

However, when using DBMS XLSX driver in the earlier maintenance release SAS® 9.3 (TS1M0), the default variable names are named incorrectly. The variable names would be off by one letter. The first variable name becomes 'B', second 'C', third 'D' ... etc.

This software bug has been documented in SAS / SUPPORT / SAMPLES & SAS NOTES Problem Note 47365 (<http://support.sas.com/kb/47/365.html>).

The code listed below is an example.

```
PROC IMPORT
    DATAFILE="C:\Biometrics Summary Excel.xlsx"
    DBMS=XLSX
    OUT=_infile_
    REPLACE;
    GETNAMES=NO;
RUN;
```

I am using maintenance release 9.03.01M0 060711 16:38 Sunday, July

Obs	B	C	D	E	F
1	Compound	MK-9999			CONFIDENTIAL
2	TT Number	16-1111			
3	Species	Rhesus Monkey			
4					
5	Dose Date	Dose Level	Dose	Animal Numbers	Summary Files
6	31-Mar-2016	Control	0	13-R353	16-1111_13-R353_Control_AB_XIVF 7.xlsx
7	31-Mar-2016	Control	0	13-R354	16-1111_13-R354_Control_AB_XIVF 7.xlsx
8	31-Mar-2016	Control	0	13-R359	16-1111_13-R359_Control_AB_XIVF 7.xlsx
9	31-Mar-2016	Control	0	13-R368	16-1111_13-R368_Control_AB_XIVF 7.xlsx
10	05-Apr-2016	3 mpkD5-1	3.1	13-R353	16-1111_13-R353_3mpk_AF_XIVF 7.xlsx
11	05-Apr-2016	3 mpkD5-1	3.1	13-R354	16-1111_13-R354_3mpk_AF_XIVF 7.xlsx
12	05-Apr-2016	3 mpkD5-1	3.1	13-R359	16-1111_13-R359_3mpk_AF_XIVF 7.xlsx
13	05-Apr-2016	3 mpkD5-1	3.1	13-R368	16-1111_13-R368_3mpk_AF_XIVF 7.xlsx
14	06-Apr-2016	3 mpkD5-2	3.2	13-R353	16-1111_13-R353_3mpk_AG_XIVF 7.xlsx
15	06-Apr-2016	3 mpkD5-2	3.2	13-R354	16-1111_13-R354_3mpk_AG_XIVF 7.xlsx

Display 3: XLSX Driver Output: SAS Version: SAS® 9.3 software (TS1M0)

The same code executed in SAS 9.4 / SAS 9.3 (TS1M1) or later release does not have this bug. The variable names are named correctly starting from A, B, C ... etc.

I am using maintenance release 9.04.01M2 072314 16:43 Sunday, July 29, 2018

Obs	A	B	C	D	E
1	Compound	MK-9999			CONFIDENTIAL
2	TT Number	16-1111			
3	Species	Rhesus Monkey			
4					
5	Dose Date	Dose Level	Dose	Animal Numbers	Summary Files
6	31-Mar-2016	Control	0	13-R353	16-1111_13-R353_Control_AB_XIVF 7.xlsx
7	31-Mar-2016	Control	0	13-R354	16-1111_13-R354_Control_AB_XIVF 7.xlsx
8	31-Mar-2016	Control	0	13-R359	16-1111_13-R359_Control_AB_XIVF 7.xlsx
9	31-Mar-2016	Control	0	13-R368	16-1111_13-R368_Control_AB_XIVF 7.xlsx
10	05-Apr-2016	3 mpkD5-1	3.1	13-R353	16-1111_13-R353_3mpk_AF_XIVF 7.xlsx

Display 4: XLSX Driver Output: SAS Version: SAS® 9.4 software / SAS® 9.3 software (TS1M1) or a later release

SOLUTION 1

DBMS EXCEL driver does not have such bug in different maintenance versions. The first variable name is always 'F1', second 'F2', third 'F3' ... etc. When there is no particular reason to use the XLSX driver, it is recommended to use EXCEL driver instead of XLSX driver in order to avoid the inconsistent variable names. An example is listed below.

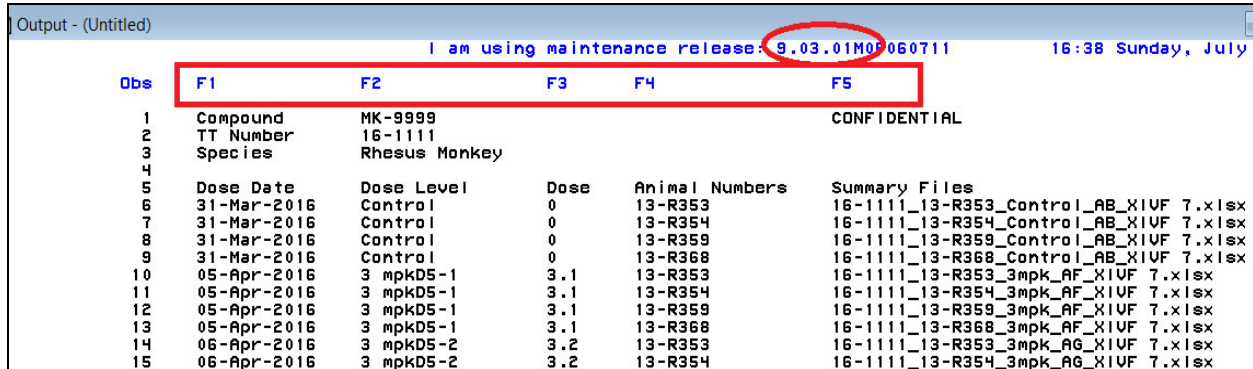
```

PROC IMPORT
    DATAFILE="C:\Biometrics Summary Excel.xlsx"
    DBMS=EXCEL
    OUT=_infile
    REPLACE;
    GETNAMES=NO;

```

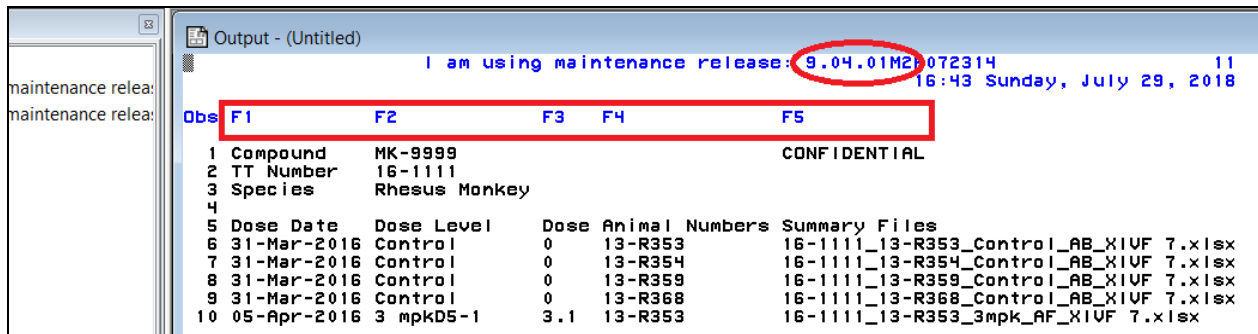
RUN;

The same SAS code executed in SAS 9.4 / SAS 9.3 (TS1M0) or later release creates the exactly same variable names.



Obs	F1	F2	F3	F4	F5
1	Compound	MK-9999			CONFIDENTIAL
2	TT Number	16-1111			
3	Species	Rhesus Monkey			
4					
5	Dose Date	Dose Level	Dose	Animal Numbers	Summary Files
6	31-Mar-2016	Control	0	13-R353	16-1111_13-R353_Control_AB_XIVF 7.xlsx
7	31-Mar-2016	Control	0	13-R354	16-1111_13-R354_Control_AB_XIVF 7.xlsx
8	31-Mar-2016	Control	0	13-R359	16-1111_13-R359_Control_AB_XIVF 7.xlsx
9	31-Mar-2016	Control	0	13-R368	16-1111_13-R368_Control_AB_XIVF 7.xlsx
10	05-Apr-2016	3 mpkD5-1	3.1	13-R353	16-1111_13-R353_3mpk_AF_XIVF 7.xlsx
11	05-Apr-2016	3 mpkD5-1	3.1	13-R354	16-1111_13-R354_3mpk_AF_XIVF 7.xlsx
12	05-Apr-2016	3 mpkD5-1	3.1	13-R359	16-1111_13-R359_3mpk_AF_XIVF 7.xlsx
13	05-Apr-2016	3 mpkD5-1	3.1	13-R368	16-1111_13-R368_3mpk_AF_XIVF 7.xlsx
14	06-Apr-2016	3 mpkD5-2	3.2	13-R353	16-1111_13-R353_3mpk_A6_XIVF 7.xlsx
15	06-Apr-2016	3 mpkD5-2	3.2	13-R354	16-1111_13-R354_3mpk_A6_XIVF 7.xlsx

Display 5: EXCEL Driver Output: SAS Version: SAS® 9.3 software (TS1M0)



Obs	F1	F2	F3	F4	F5
1	Compound	MK-9999			CONFIDENTIAL
2	TT Number	16-1111			
3	Species	Rhesus Monkey			
4					
5	Dose Date	Dose Level	Dose	Animal Numbers	Summary Files
6	31-Mar-2016	Control	0	13-R353	16-1111_13-R353_Control_AB_XIVF 7.xlsx
7	31-Mar-2016	Control	0	13-R354	16-1111_13-R354_Control_AB_XIVF 7.xlsx
8	31-Mar-2016	Control	0	13-R359	16-1111_13-R359_Control_AB_XIVF 7.xlsx
9	31-Mar-2016	Control	0	13-R368	16-1111_13-R368_Control_AB_XIVF 7.xlsx
10	05-Apr-2016	3 mpkD5-1	3.1	13-R353	16-1111_13-R353_3mpk_AF_XIVF 7.xlsx

Display 6: EXCEL Driver Output: SAS Version: SAS® 9.4 software / SAS® 9.3 software (TS1M1) or a later release

SOLUTION 2

Utilize automatic variable SYSVLONG to direct SAS code using the correct procedure per different SAS version or maintenance release.

The code listed below corrects the variable names. If the computer user has SAS 9.3 (TS1M0), this code will be executed. For the same code in SAS 9.3 (TS1M1), later maintenance release or SAS 9.4, this step will be skipped. An example is listed below.

```

%macro fixdata;
    %if %sysfunc(substr(&sysvlong,1,9)) = 9.03.01M0 %then %do;
        data _infile;
            set _infile (rename=(B=A C=B D=C E=D F=E));
        run;

        proc print data=_infile(obs=10);
            title1 "I am using maintenance release: &sysvlong";
            title2 "data is fixed";
        run;
    %end;
%mend;

```

ISSUE 2: MISUSE OF MIXED STATEMENT IN DBMS=XLSX

Statement MIXED=YES is used frequently in IMPORT procedure to convert both character and numeric data in the same column to character data only. It could happen that DBMS=XLSX works correctly when it is used in SAS 9.3, but generates error messages in SAS 9.4. See examples below.

```
PROC IMPORT DATAFILE="C:\Biometrics Summary Excel.xlsx"
  DBMS=XLSX
  OUT=_infile
  REPLACE;
  MIXED=YES;
RUN;
```

```
139 %put I am using maintenance release: &sysvlong;
140 I am using maintenance release: 9.03.01M0P060711
141 PROC IMPORT DATAFILE="C:\Huei-Ling\ProcImportDBMS\data\Biometrics Summary Excel.xlsx"
142           DBMS=XLSX OUT=_infile REPLACE;
143           GETNAMES=NO;
144           MIXED=YES;
145 RUN;

NOTE: The import data set has 99 observations and 5 variables.
NOTE: WORK._INFILES data set was successfully created.
NOTE: PROCEDURE IMPORT used (Total process time):
      real time           0.03 seconds
      cpu time            0.01 seconds
```

Display 7: XLSX Driver Output: there is no error message in SAS® 9.3

```
81 %put I am using maintenance release: &sysvlong;
82 I am using maintenance release: 9.04.01M2P072314
83 PROC IMPORT DATAFILE="C:\Huei-Ling\ProcImportDBMS\data\Biometrics Summary Excel.xlsx"
84           DBMS=XLSX OUT=_infile REPLACE;
NOTE: The previous statement has been deleted.
85           GETNAMES=NO;
86           MIXED=YES;
      -----
      180
ERROR 180-322: Statement is not valid or it is used out of proper order.
87 RUN;

NOTE: The SAS System stopped processing this step because of errors.
NOTE: PROCEDURE IMPORT used (Total process time):
      real time           0.21 seconds
      cpu time            0.03 seconds
```

Display 8: XLSX Driver Output: same code in SAS® 9.4 creates error message

The reason for the error message according to “SAS/ACCESS® 9.4 Interface to PC Files: Reference, Fourth Edition” is in SAS 9.4 both XLSX and XLS formats assume MIXED=YES. There is no option to select MIXED=YES or NO. Furthermore, XLSX only supports options GETNAMES, RANGE, and SHEET.

Table 6.8 Available Statements for Importing and Exporting Excel Files Using DBMS=XLS and DBMS=XLSX

DBMS= Identifier	Option	Valid Value	Default Value	PROC IMPORT	PROC EXPORT
XLS	ENDCOL	Last column for data	Last column that contains data	Yes	No
	ENDNAMEROW	Last row for variable names	Same as NAMEROW	Yes	No
	ENDROW	Last row for data	Last row that contains data	Yes	No
	GETNAMES	Yes No	Yes	Yes	No
	NAMEROW	First row for variable names	First row that contains variable names	Yes	No
	NEWFILE	Yes No	No	No	Yes
	PUTNAMES	Yes No	Yes	No	Yes
	RANGE	name sheet \$u:l:r	First row	Yes	No
	SHEET	Sheet name	First sheet	Yes	Yes
	STARTCOL	First column for data	Last column that contains data	Yes	No
	STARTROW	First row for data	First row that contains data	Yes	No
XLSX	GETNAMES	Yes No	Yes	Yes	No
	RANGE	name sheet \$u:l:r	First row	Yes	No
	SHEET	Sheet name	First sheet	Yes	Yes

In SAS® 9.4, an error message will show if MIXED= option is used with DBMS=XLSX.

However, in SAS® 9.3, though MIXED= option is not supported by DBMS=XLSX, no error message is produced.

The logic is that Mixed=YES examines the first eight rows by default in the Excel data file. When the value is mixed with both character and numeric value, the column is converted to character variable. Actually MIXED= is only compatible with the MS Jet/Excel engine, therefore only DBMS=EXCEL (SAS EXCEL LIBNAME ENGINE) supports MIXED option. DBMS=XLSX does not support this feature.

SOLUTION

In PROC IMPORT procedure with DBMS=XLSX option, the MIXED option should be removed. Although SAS 9.3 does not produce an error message, MIXED= should not be used with DBMS=XLSX in a program, so that the same code can be reused in later versions.

CONCLUSION

Procedure PROC IMPORT is a common tool used to convert Excel data file to SAS dataset. The two possibilities which could generate inconsistent SAS outputs from the procedure PROC IMPORT were explained in this paper. It was observed the discrepancies were due to either different SAS versions or different maintenance releases used on different PCs. Detailed discussions on the causes and the solutions for inconsistent outputs were provided. It was concluded that the automatic variable SYSVLONG is a practical and effective tool to enhance the robustness of an automated SAS program.

REFERENCES

- <http://support.sas.com/kb/47/365.html>
- <http://support.sas.com/kb/13/526.html>
- <http://documentation.sas.com/?docsetId=acpcref&docsetTarget=p01fkW2p1ly97hn123n2pvfwi6ee.htm&docsetVersion=9.4&locale=en>
- SAS/ACCESS® 9.3 Interface to PC Files Reference
- SAS/ACCESS® 9.4 Interface to PC Files Reference

ACKNOWLEDGMENTS

The authors would like to thank the management teams for their advice on this paper/presentation.

CONTACT INFORMATION

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

Huei-Ling Chen
c/o Merck & Co., Inc.
126 Lincoln Avenue
P.O. Box 2000
Rahway, NJ 07065
Phone: 732-594-2287
e-mail: huei-ling_chen@merck.com

Chao-Min Hoe
c/o Merck & Co., Inc.
770 Sumneytown Pike
West Point, PA. 19486
Phone: 215-652-5637
e-mail: hoe_chao-min@merck.com

TRADEMARK

SAS and all other SAS Institute Inc. products or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.