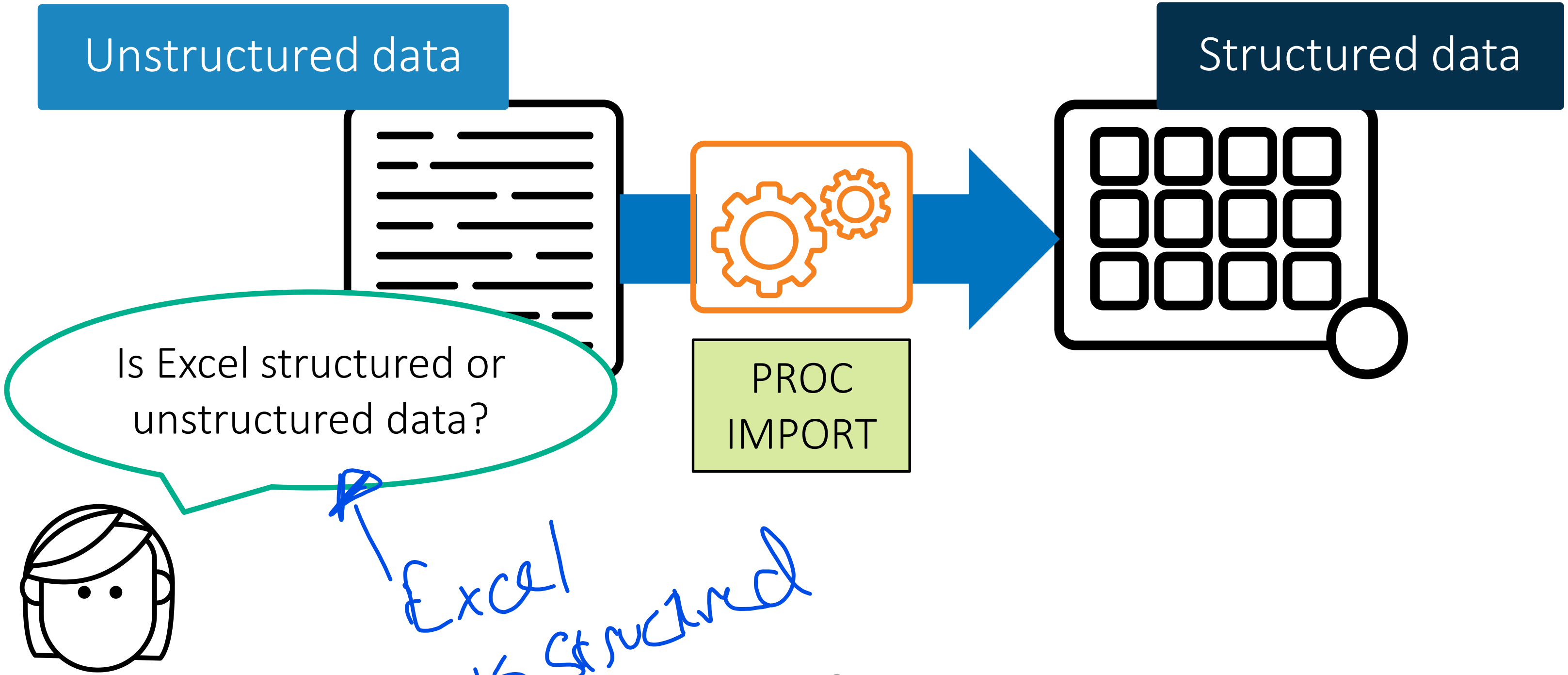
SAS Lesson 04

Importing Unstructured Data



Unstructured data

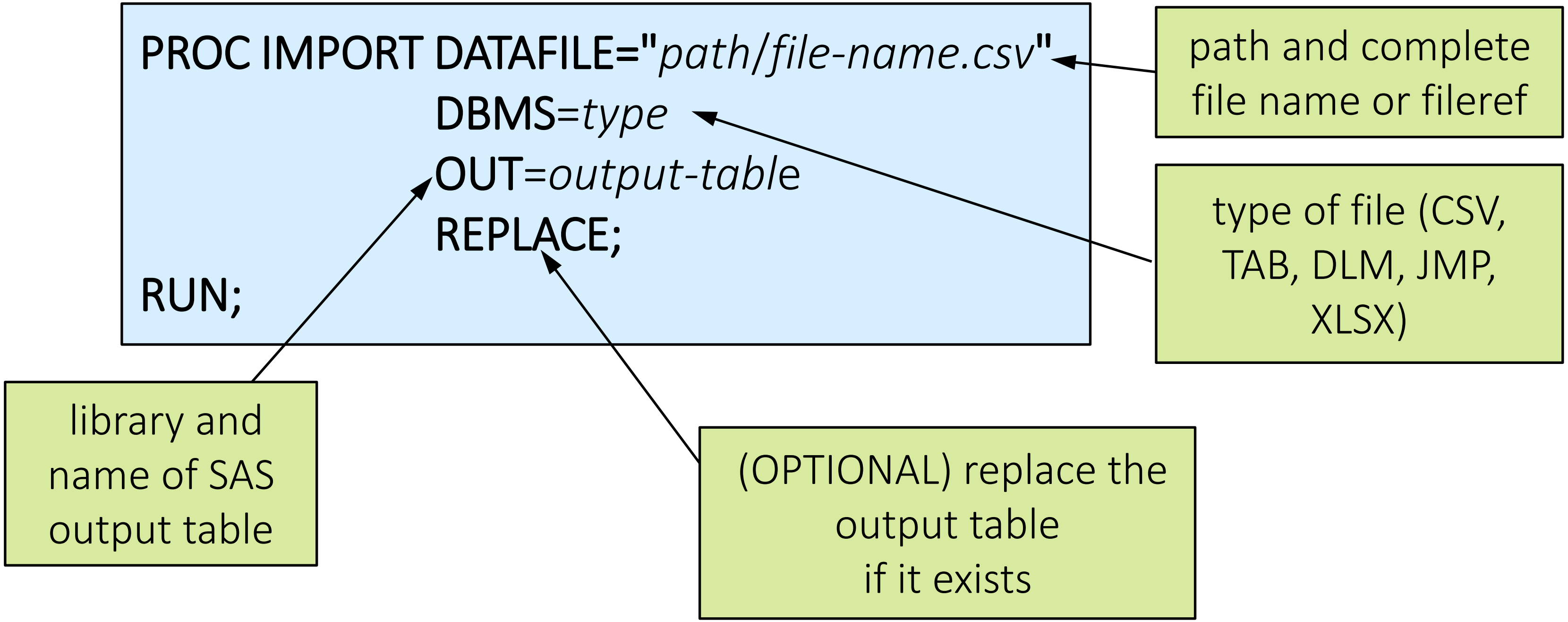
PROC IMPORT

Structured data

Is Excel structured or unstructured data?

Excel is structured

Importing Unstructured Data – General Form



```
PROC IMPORT DATAFILE="path/file-name.csv"
                DBMS=type
                OUT=output-table
                REPLACE;
RUN;
```

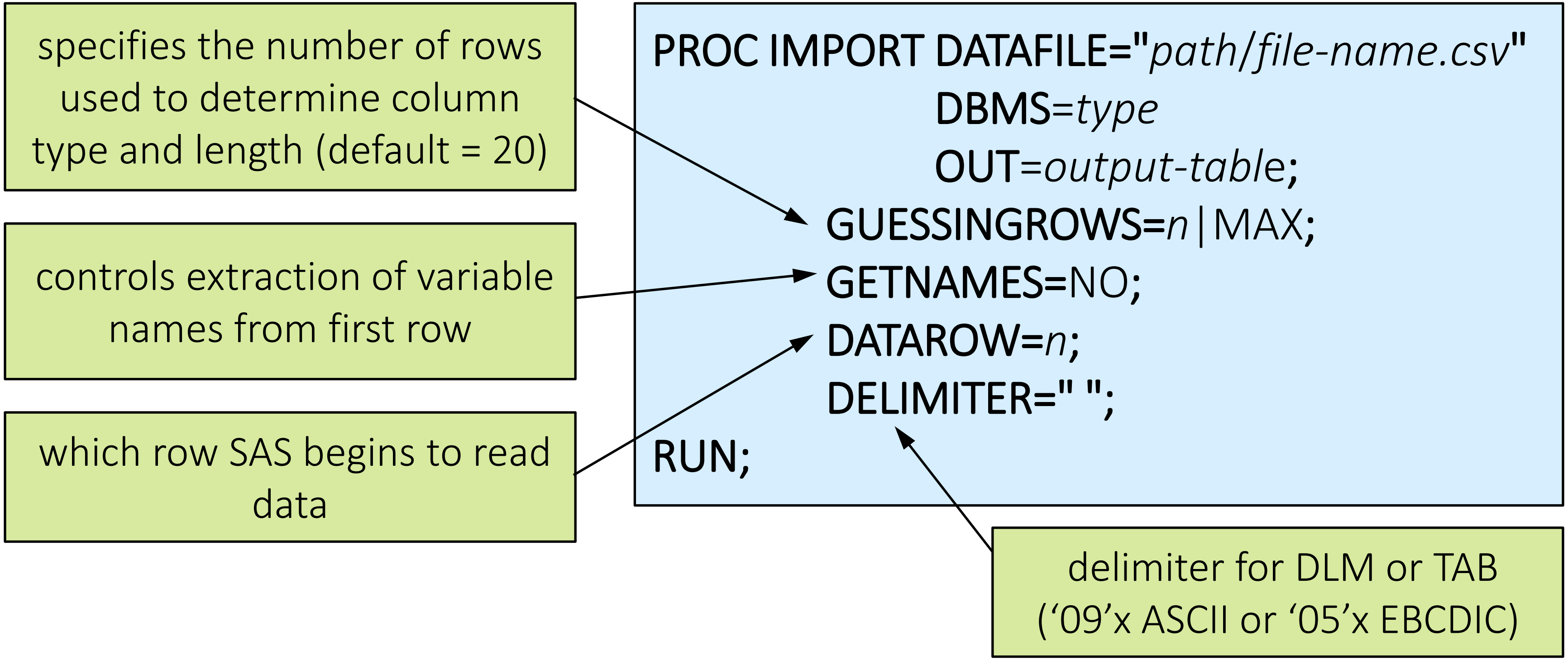
path and complete file name or fileref

type of file (CSV, TAB, DLM, JMP, XLSX)

library and name of SAS output table

(OPTIONAL) replace the output table if it exists

Importing Unstructured Data – Optional Statements



```
PROC IMPORT DATAFILE="path/file-name.csv"
                DBMS=type
                OUT=output-table;
          GUESSINGROWS=n|MAX;
          GETNAMES=NO;
          DATAROW=n;
          DELIMITER=" ";
RUN;
```

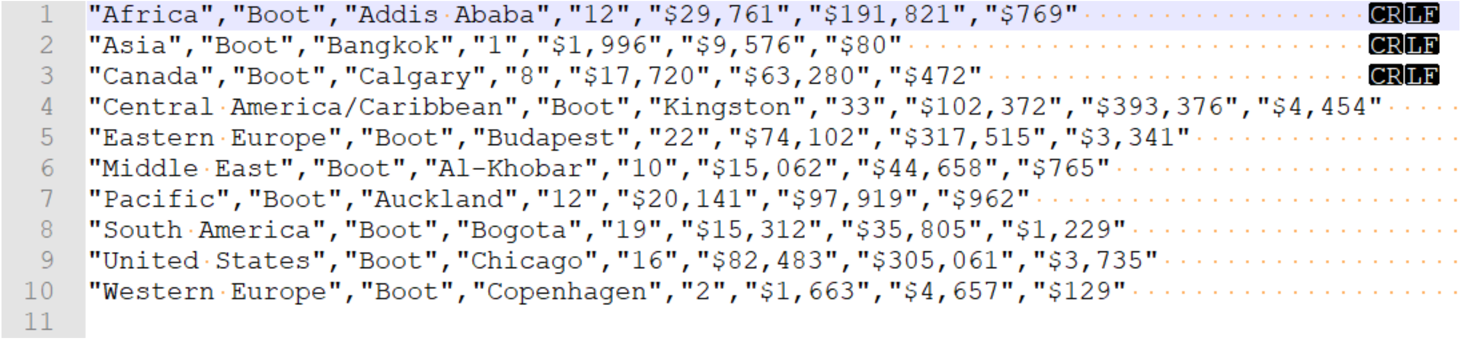
specifies the number of rows used to determine column type and length (default = 20)

controls extraction of variable names from first row

which row SAS begins to read data

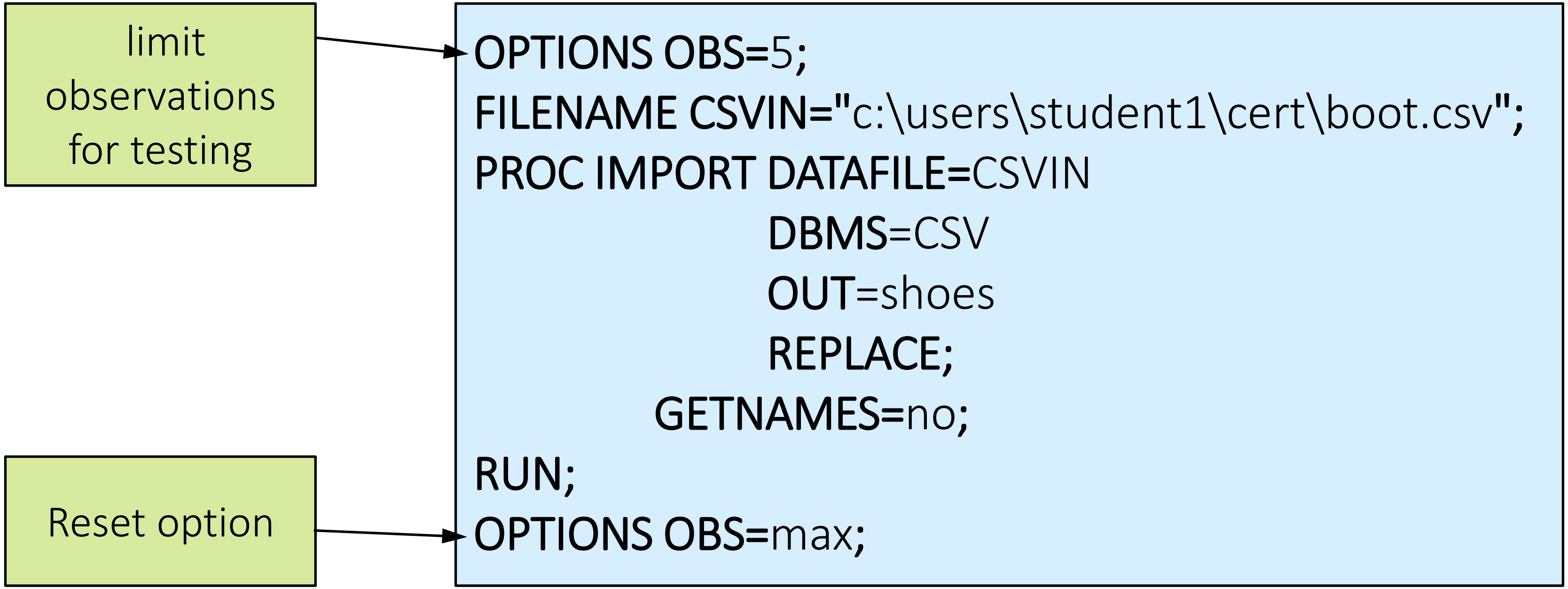
delimiter for DLM or TAB (‘09’x ASCII or ‘05’x EBCDIC)

Importing a Comma-Delimited (CSV) File



```
"Africa","Boot","Addis Ababa","12","$29,761","$191,821","$769"
"Asia","Boot","Bangkok","1","$1,996","$9,576","$80"
"Canada","Boot","Calgary","8","$17,720","$63,280","$472"
"Central America/Caribbean","Boot","Kingston","33","$102,372","$393,376","$4,454"
"Eastern Europe","Boot","Budapest","22","$74,102","$317,515","$3,341"
"Middle East","Boot","Al-Khobar","10","$15,062","$44,658","$765"
"Pacific","Boot","Auckland","12","$20,141","$97,919","$962"
"South America","Boot","Bogota","19","$15,312","$35,805","$1,229"
"United States","Boot","Chicago","16","$82,483","$305,061","$3,735"
"Western Europe","Boot","Copenhagen","2","$1,663","$4,657","$129"
```

Importing a Comma-Delimited (CSV) File



limit observations for testing

Reset option

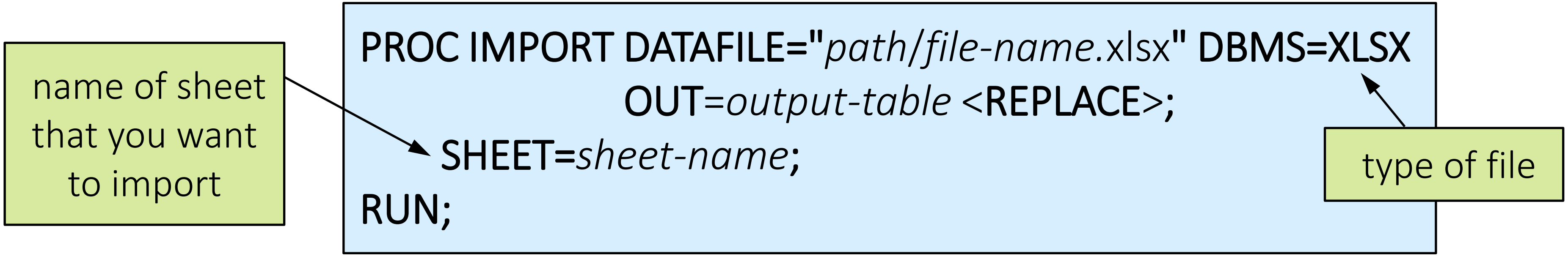
```
OPTIONS OBS=5;
FILENAME CSVIN="c:\users\student1\cert\boot.csv";
PROC IMPORT DATAFILE=CSVIN
            DBMS=CSV
            OUT=shoes
            REPLACE;
        GETNAMES=no;
RUN;
OPTIONS OBS=max;
```

Importing a Tab-Delimited File

This demonstration illustrates importing a tab-delimited file and creating a new SAS table using
PROC IMPORT.

07-Import.sas

Importing an Excel File



name of sheet that you want to import

PROC IMPORT DATAFILE="*path/file-name*.xlsx" DBMS=XLSX
OUT=*output-table* <REPLACE>;
SHEET=*sheet-name*;
RUN;

type of file

```
filename xlin "c:\users\student1\cert\boots.xlsx" ;
proc import datafile= xlin
            dbms=xlsx
            out=work.bootsales replace;
    sheet=boot;
run;
```

Importing an Excel File

This demonstration illustrates importing a file from Excel and creating a new SAS table using
PROC IMPORT.

07-Import.sas

Discussion

What is the difference between using the XLSX LIBNAME engine and PROC IMPORT to read Excel data in a SAS program?

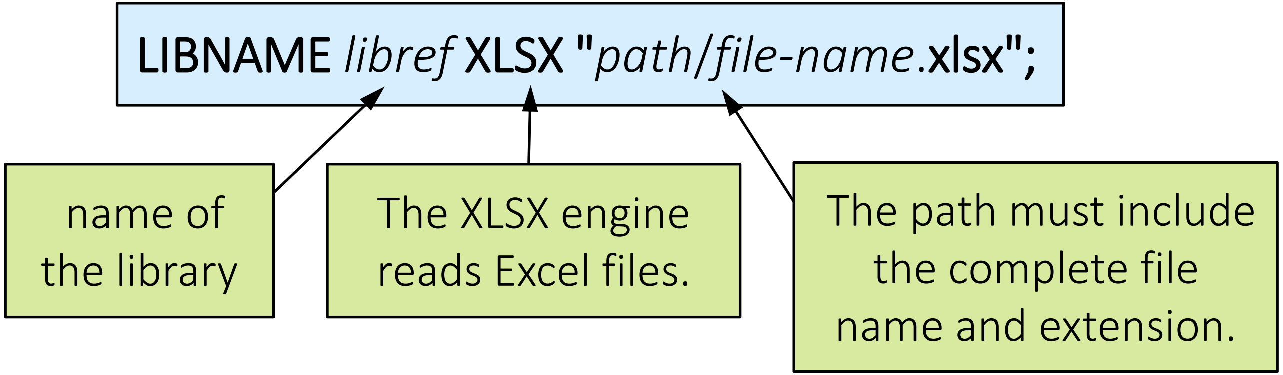
SAS LIBNAME Engines for Microsoft Excel

Comparison of the SAS LIBNAME Engines That Access Microsoft Excel Data

Feature	EXCEL LIBNAME Engine	PCFILES LIBNAME Engine	XLSX LIBNAME Engine
Host support	Microsoft Windows	Windows, UNIX	Windows, UNIX, Studio
Requires the SAS PC Files Server	No	Yes	No
Requires Microsoft Access Database Engine (ACE)	Yes *	Yes	No
Supports SAS LIBNAME options	Yes	Yes	Limited
Supports SAS data set options	Yes	Yes	Limited
Supports SAS SQL procedure and pass-through	Yes	Yes	No
Reads data support for file types	.xlsx, .xlsb, .xlsm, .xls	.xlsx, .xlsb, .xlsm, .xls	.xlsx
Creates data support for file types (new table)	.xlsx, .xlsb, .xls	xlsx, .xlsb, .xls	.xlsx
Updates data support for file types	.xlsx, .xlsb, ..xls	.xlsx, .xlsb, .xls	.xlsx

* Requires bit consistency

Using a Library to Read Excel Files (Review)



LIBNAME *libref* XLSX "*path/file-name*.xlsx";

name of the library

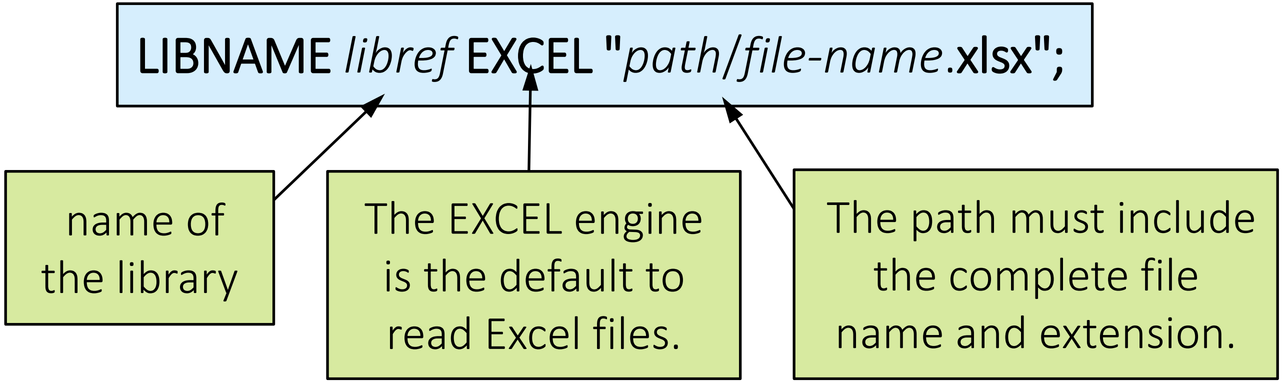
The XLSX engine reads Excel files.

The path must include the complete file name and extension.

```
libname xlclass xlsx "s:/workshop/data/class.xlsx";
```

The XLSX engine requires a license for SAS/ACCESS Interface to PC Files.

Using the EXCEL Engine to Read Excel Files (PC SAS)



LIBNAME *libref* EXCEL "*path/file-name*.xlsx";

name of the library

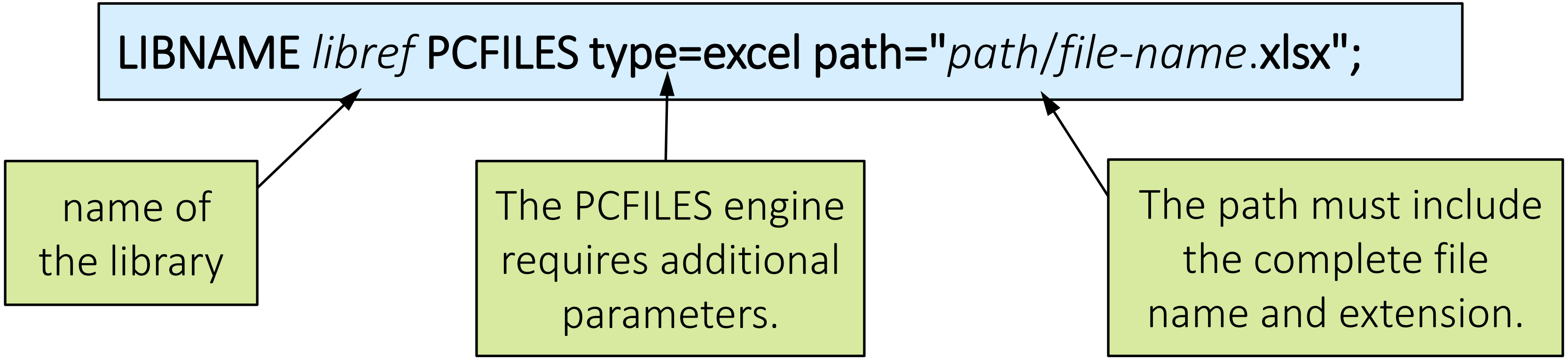
The EXCEL engine is the default to read Excel files.

The path must include the complete file name and extension.

```
libname xlclass excel "s:/workshop/data/class.xlsx";
```

The EXCEL engine requires a license for SAS/ACCESS Interface to PC Files and bit agreement.

Using PCFILES Server to Read Excel Files (PC SAS)



LIBNAME *libref* PCFILES type=excel path="*path/file-name*.xlsx";

name of the library

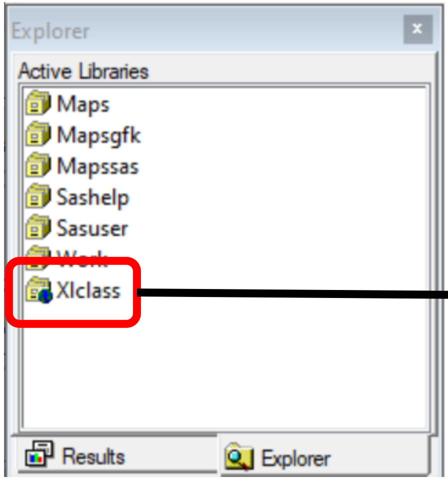
The PCFILES engine requires additional parameters.

The path must include the complete file name and extension.

```
libname xlclass pcfiles type=excel
path="s:/workshop/data/class.xlsx";
```

Use PCFILES to access Excel workbooks when the ACE and OS have a bit mismatch. Requires installation of PCFILES server service on Windows.

SAS Explorer Window

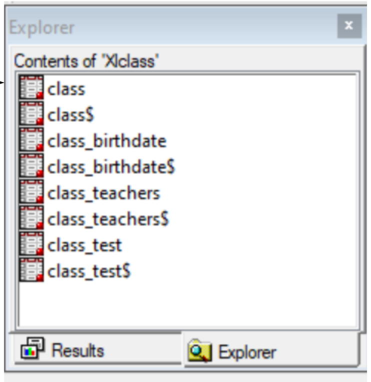


Explorer

Active Libraries

- Maps
- Mapsgfk
- Mapssas
- Sashelp
- Sasuser
- Work
- Xlclass

Each worksheet in the Excel workbook is treated as though it is a SAS data set.



Explorer

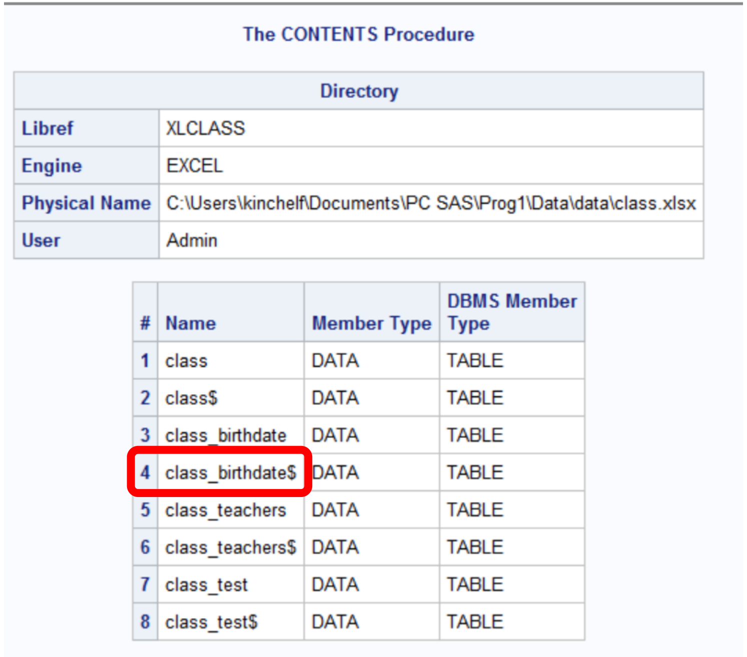
Contents of 'Xlclass'

- class
- class$
- class_birthdate
- class_birthdate$
- class_teachers
- class_teachers$
- class_test
- class_test$

Worksheet names appear with a dollar sign at the end of the name with EXCEL and PCFILES engines.

The CONTENTS Procedure

```
proc contents data=xlclass._all_ nods;
run;
```



The CONTENTS Procedure

Directory	
Libref	XLCLASS
Engine	EXCEL
Physical Name	C:\Users\kinchelf\Documents\PC SAS\Prog1\Data\data\class.xlsx
User	Admin

#	Name	Member Type	DBMS Member Type
1	class	DATA	TABLE
2	class$	DATA	TABLE
3	class_birthdate	DATA	TABLE
4	class_birthdate$	DATA	TABLE
5	class_teachers	DATA	TABLE
6	class_teachers$	DATA	TABLE
7	class_test	DATA	TABLE
8	class_test$	DATA	TABLE

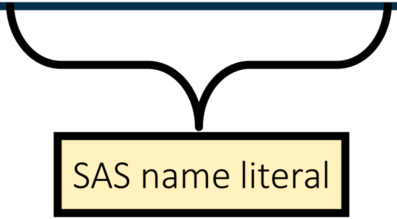
SAS Name Literals

By default, special characters such as the $ are not allowed in data set names.

SAS name literals enable special characters to be included in data set names.

A *SAS name literal* is a name token that is expressed as a string within quotation marks, followed by the letter n.

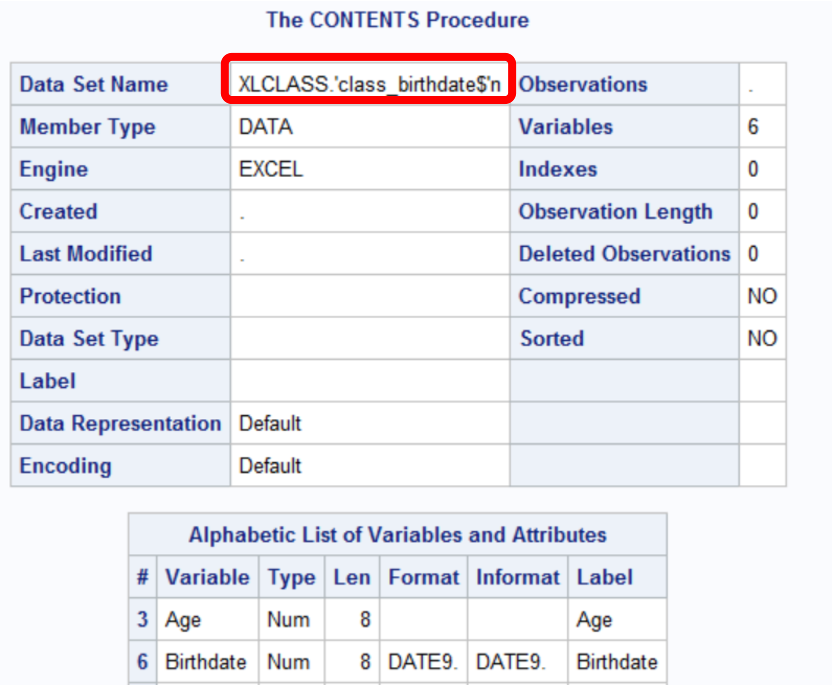
```
proc contents data=xlclass.'class_birthdate$'n
```



SAS name literal

The CONTENTS Procedure

```
proc contents data=xlclass.'class_birthdate$'n;
run;
```



The CONTENTS Procedure

Data Set Name	XLCLASS.'class_birthdate$'n	Observations	.
Member Type	DATA	Variables	6
Engine	EXCEL	Indexes	0
Created	.	Observation Length	0
Last Modified	.	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	Default		
Encoding	Default		

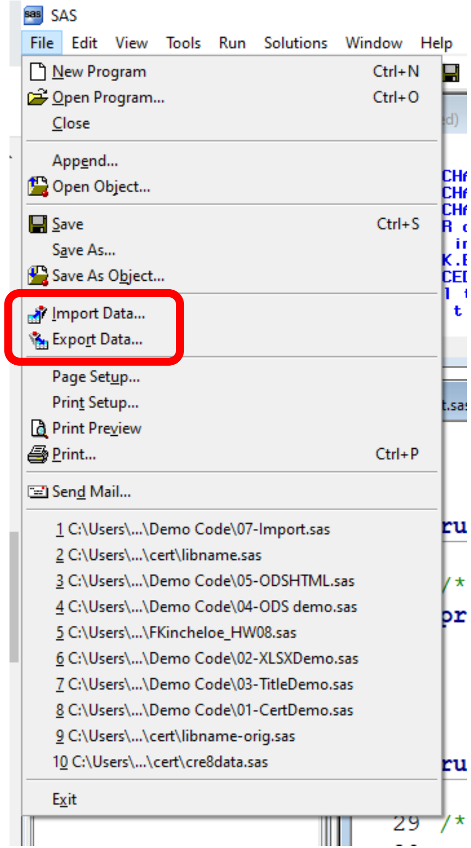
Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Informat	Label
3	Age	Num	8			Age
6	Birthdate	Num	8	DATE9.	DATE9.	Birthdate

Exporting Results

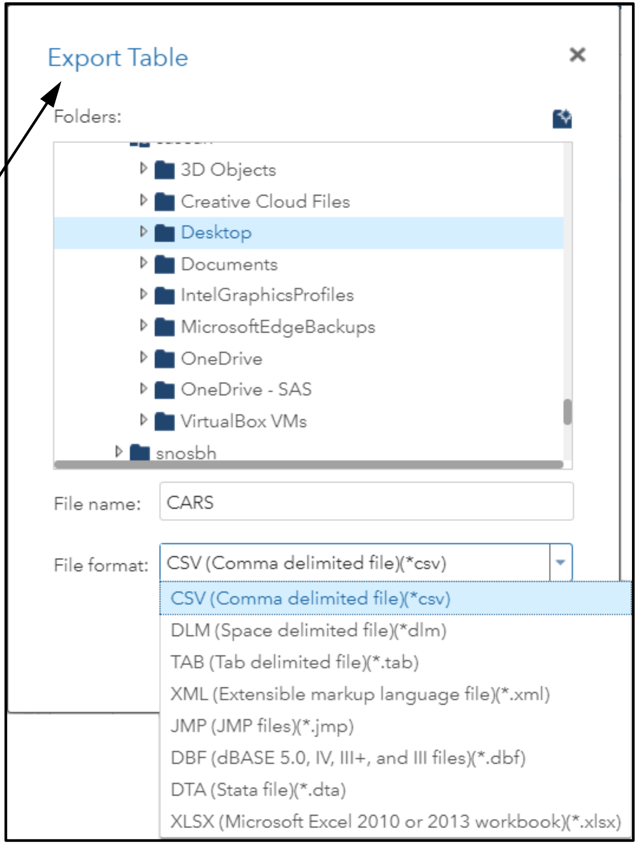
Exporting Data – Prep Guide page 356

Exporting Data Using Point-and-Click Tools


SAS
File Edit View Tools Run Solutions Window Help
New Program Ctrl+N
Open Program... Ctrl+O
Close
Append...
Open Object...
Save Ctrl+S
Save As...
Save As Object...
Import Data...
Export Data...
Page Setup...
Print Setup...
Print Preview
Print... Ctrl+P
Send Mail...
1 C:\Users\...\Demo Code\07-Import.sas
2 C:\Users\...\cert\libname.sas
3 C:\Users\...\Demo Code\05-ODSHTML.sas
4 C:\Users\...\Demo Code\04-ODS demo.sas
5 C:\Users\...\FKincheloe_HW08.sas
6 C:\Users\...\Demo Code\02-XLSXDemo.sas
7 C:\Users\...\Demo Code\03-TitleDemo.sas
8 C:\Users\...\Demo Code\01-CertDemo.sas
9 C:\Users\...\cert\libname-orig.sas
10 C:\Users\...\cert\cre8data.sas
Exit

Right-click a table in SAS Studio and select Export.

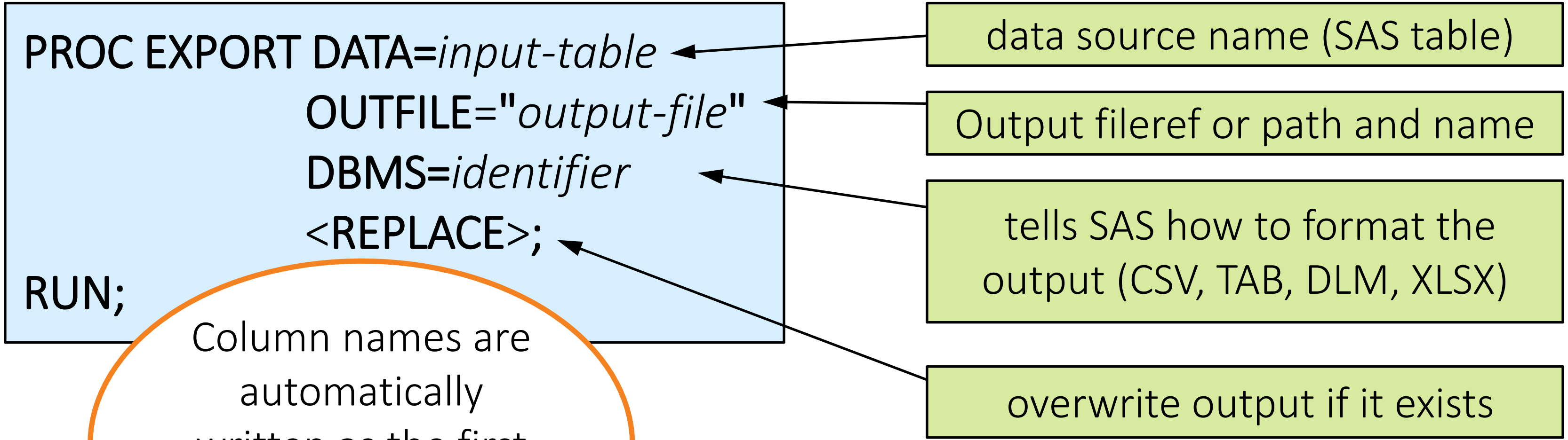
Import wizard is under Tasks and Utilities


Export Table
Folders:
3D Objects
Creative Cloud Files
Desktop
Documents
IntelGraphicsProfiles
MicrosoftEdgeBackups
OneDrive
OneDrive - SAS
VirtualBox VMs
snosbh
CSV (Comma delimited file)(*csv)
DLM (Space delimited file)(*dlm)
TAB (Tab delimited file)(*.tab)
XML (Extensible markup language file)(*.xml)
JMP (JMP files)(*.jmp)
DBF (dBASE 5.0, IV, III+, and III files)(*.dbf)
DTA (Stata file)(*.dta)
XLSX (Microsoft Excel 2010 or 2013 workbook)(*.xlsx)

File name: CARS

File format: CSV (Comma delimited file)(*csv)

Exporting Data Using Code



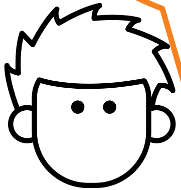
```
PROC EXPORT DATA=input-table
                 OUTFILE="output-file"
                 DBMS=identifier
                 <REPLACE>;
RUN;
```

data source name (SAS table)

Output fileref or path and name

tells SAS how to format the output (CSV, TAB, DLM, XLSX)

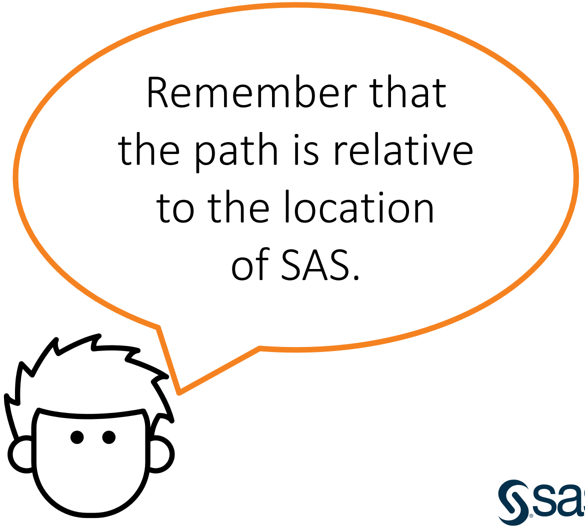
overwrite output if it exists



Column names are automatically written as the first row of the output file.

Exporting Data Using Code

```
proc export data=sashelp.cars
    outfile= "/folders/myfolders/output/cars.txt"
    dbms=tab replace;
run;
```



Exporting Data with a LIBNAME Engine

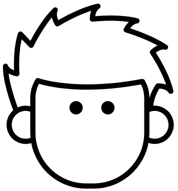
```
libname myxl xlsx "/folders/myfolders/output/cars.xlsx";

data myxl.asiacars;
    set sashelp.cars;
    where origin='Asia';
run;

libname myxl clear;
```

defines a library to the Microsoft Excel workbook that you are creating

This code extracts data and writes it to the **cars** workbook on a tab named **asiacars**.



Exporting Data to an Excel Workbook

This demonstration illustrates using the XLSX LIBNAME engine and PROC Export to export SAS tables to multiple worksheets in an Excel workbook.

Discussion

What is the difference between using ODS Excel and PROC EXPORT to create Excel data in a SAS program?

Accessing Data

Creating data sets from SAS tables

Using a SAS Data Set as Input

```
data men50.males;
   set cert.admit;
   where sex= 'M' and
         age > 50;
run;
```

DATA *output-SAS-data-set***;**
    SET *input-SAS-data-set***;**
    WHERE *WHERE-expression***;**
RUN;

DATA Statement

The *DATA statement* begins a DATA step and provides the name of the SAS data set to create.

```
data men50.males;
   set cert.admit;
   where sex= 'M' and
            age > 50;
run;
```

DATA *output-SAS-data-set*;

A DATA step can create temporary or permanent data sets.

The rules for SAS variable names also apply to data set names.

SET Statement

The *SET statement* reads observations from an existing SAS data set for further processing in the DATA step.

```
data men50.males;
   set cert.admit;
   where sex= 'M' and
            age > 50;
run;
```

SET *input-SAS-data-set*;

- The SET statement reads all observations and all variables from the input data set.
- Observations are read sequentially, one at a time.
- The SET statement can read temporary or permanent data sets.

WHERE Statement

The *WHERE statement* selects observations from a SAS data set that meet a particular condition.

WHERE *WHERE-expression*;

```
data men50.males;
   set cert.admit;
   where sex= 'M' and
         age > 50;
run;
```

The variables named in the WHERE expression must exist in the input SAS data set.

Multiple Choice Poll

Considering this DATA step, which statement is true?

```
data us;
   set orion.sales;
   where Country='US';
run;
```

a.It reads a temporary data set and creates a permanent data set.

[✓ b.]It reads a permanent data set and creates a temporary data set.

c.It contains a syntax error and does not execute.

d.It does not execute because you cannot work with permanent and temporary data sets in the same step.

Multiple Choice Poll – Correct Answer

Considering this DATA step, which statement is true?

a. It reads a temporary data set and creates a permanent data set.

[✓ b.] It reads a permanent data set and creates a temporary data set.

c. It contains a syntax error and does not execute.

d. It does not execute because you cannot work with permanent and temporary data sets in the same step.

```
data us;                          /* Create a temporary data set */
   set orion.sales;    /* Read a permanent data set */
   where Country='US';
run;
```

Lesson Quiz

9. What does this code do?

```
proc import datafile="d:/collect817/bird_count.csv"
            dbms=csv out=bird817 replace;
run;
```

a. It creates a SAS table named **Bird817** in the **Work** library from the CSV file **bird_count** and replaces **Bird817** whenever the CSV file is updated.

b. It creates a SAS table named **Bird817** in the **Work** library from the CSV file **bird_count**.

c. It uses the CSV engine to directly read the data file **bird_count.csv**.

9. What does this code do?

```
proc import datafile="d:/collect817/bird_count.csv"
            dbms=csv out=bird817 replace;
run;
```

a. It creates a SAS table named **Bird817** in the **Work** library from the CSV file **bird_count** and replaces **Bird817** whenever the CSV file is updated.

[✓ b.] It creates a SAS table named **Bird817** in the **Work** library from the CSV file **bird_count**.

c. It uses the CSV engine to directly read the data file **bird_count.csv**.

4. Which statement disassociates the **sales** libref?

```
libname sales xlsx 'c:\mydata\midyear.xlsx';
```

a. **`libname sales end;`**

b. **`libname sales clear;`**

c. **`libname sales close;`**

d. **`libname sales disassociate;`**

4. Which statement disassociates the **sales** libref?

```
libname sales xlsx 'c:\mydata\midyear.xlsx';
```

a. `libname sales end;`

[✓ b.] `libname sales clear;`

c. `libname sales close;`

d. `libname sales disassociate;`