

# STA 311 Statistical Computing & Data Management

Instructor: Cheng Peng, Ph.D.  
Department of Mathematics  
West Chester University  
West Chester, PA 19383

Office: 25 University Avenue, RM 111  
Phone: 610.436.2369  
Email: [cpeng@wcupa.edu](mailto:cpeng@wcupa.edu)

# Topics

- ☐ Review: Input Styles, OPTIONS, LENGTH, FORMAT/INFORMAT, etc.
- ☐ SET statement – Reading existing SAS data sets
- ☐ Reading in other format data generated from other applications-WIZARD
- ☐ Reading in data from other applications – PROC IMPORT.
- ☐ Export SAS data set to a CSV file

# Three Basic Input Styles: Examples

```
DATA temp;
  input subj 1-4 name $ 6-23 gender 25 height 27-28 weight 30-32;
  CARDS;
1024 Alice Smith      1 65 125
1167 Maryann White   1 68 140
1168 Thomas Jones    2 68 190
1201 Benedictine Arnold 2 68 190
1302 Felicia Ho      1 63 115
;
RUN;
```

**Column Input**

```
PROC PRINT data=temp;
  title 'Output dataset: TEMP';
RUN;
```

```
DATA temp;
  input subj name $ gender height weight;
  CARDS;
1024 Alice 1 65 125
1167 Maryann 1 68 140
1168 Thomas 2 68 190
1201 Benedictine . 68 190
1302 Felicia 1 63 115
;
RUN;
```

**List Input**

```
PROC PRINT data=temp NOOBS;
  title 'Output dataset: TEMP';
RUN;
```

```
DATA temp;
  input @1 subj 4.
        @6 f_name $11.
        @18 l_name $6.
        +3 height 2.
        +5 wt_date mmddyy8.
        +1 calorie comma5.;
  format wt_date mmddyy8. calorie comma5.;
  DATALINES;
1024 Alice      Smith  1 65 125 12/1/95  2,036
1167 Maryann    White  1 68 140 12/01/95 1,800
1168 Thomas     Jones  2   190 12/2/95  2,302
1201 Benedictine Arnold 2 68 190 11/30/95 2,432
1302 Felicia    Ho     1 63 115 1/1/96   1,972
;
RUN;
```

**Formatted Input**

```
PROC PRINT data = temp;
  title 'Output dataset: TEMP';
  id subj;
RUN;
```

# Formats and Informats of SAS Variables

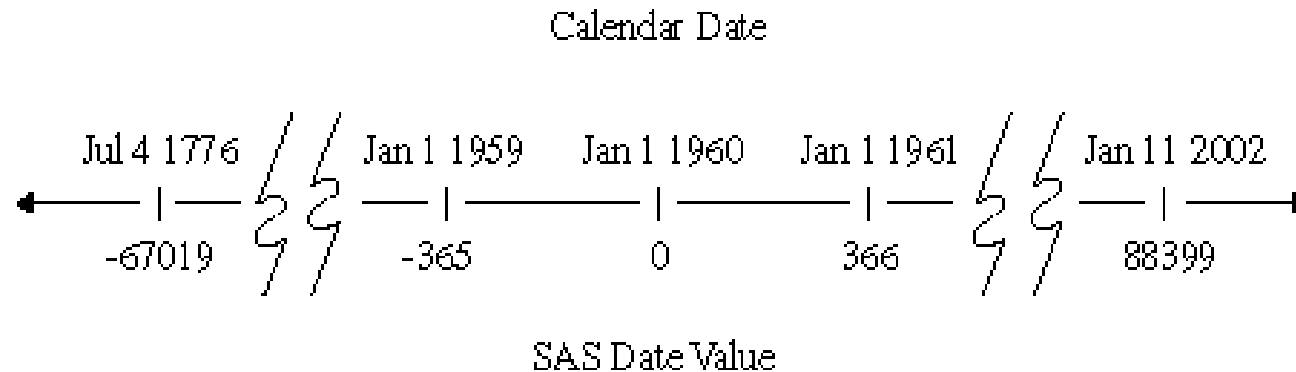
- **Informats** is used to tell SAS how to **read a variable** whereas **Formats** is used to tell SAS how to **display or write values** of a variable.
- **Informats** is basically used when you read or import data from either an external file (Text/Excel/CSV) or read in sample data which was created using CARDS/DATALINES statement. It is also used when you create a new variable in a dataset.
- **Formats** can be used in both Data Steps and PROC Steps *whereas Informat can be used only in Data Steps.*

## Setting the Length of a Variable

- ❑ The default length for character and numeric variables is 8 bytes in SAS.
- ❑ SAS uses exactly one byte for one character! This means that if the value of a character variable has more than 8 characters, SAS only keeps the first 8 characters (including the white space if any) and truncate the rest.
- ❑ However, for a numeric variable SAS variable, 8 bytes can store a number with up to 16 digits. In other words, the default length of numeric variable is 16 digits.
- ❑ It is important to note that the minimum length of a numeric is 3 bytes. It does not mean it cannot store a numeric value lower than 3 digits. It can store values of 1 or 2 digits.
- ❑ The maximum length of any character value in SAS is 32,767 bytes!

## A First Glance of SAS Dates: Value and Formats

**A SAS date is stored as a numerical value - 1/1/1960 00:00:00 as ZERO**



**A SAS Date has different formats, so does SAS time!**

## Summary: SAS Data Sets

**SAS Data Set** - a binary formatted representation of the input data set stored in such a way that future SAS programs do not need to input the data in again.

**Temporary SAS Data Sets** - created and remain in working memory for the SAS session, but disappear when the SAS session ends. Fine for small to moderate size, simple input programs.

**Permanent SAS Data Sets** - created in one SAS session but stored on disk for later reuse. Convenient for large or complex input data sets that may require multiple analysis steps. Reduces time and computer resources.

**LIBNAME** statement - identifies to the SAS program where the previously created SAS data set is located.

### Reading a single existing SAS data set using SET Statement

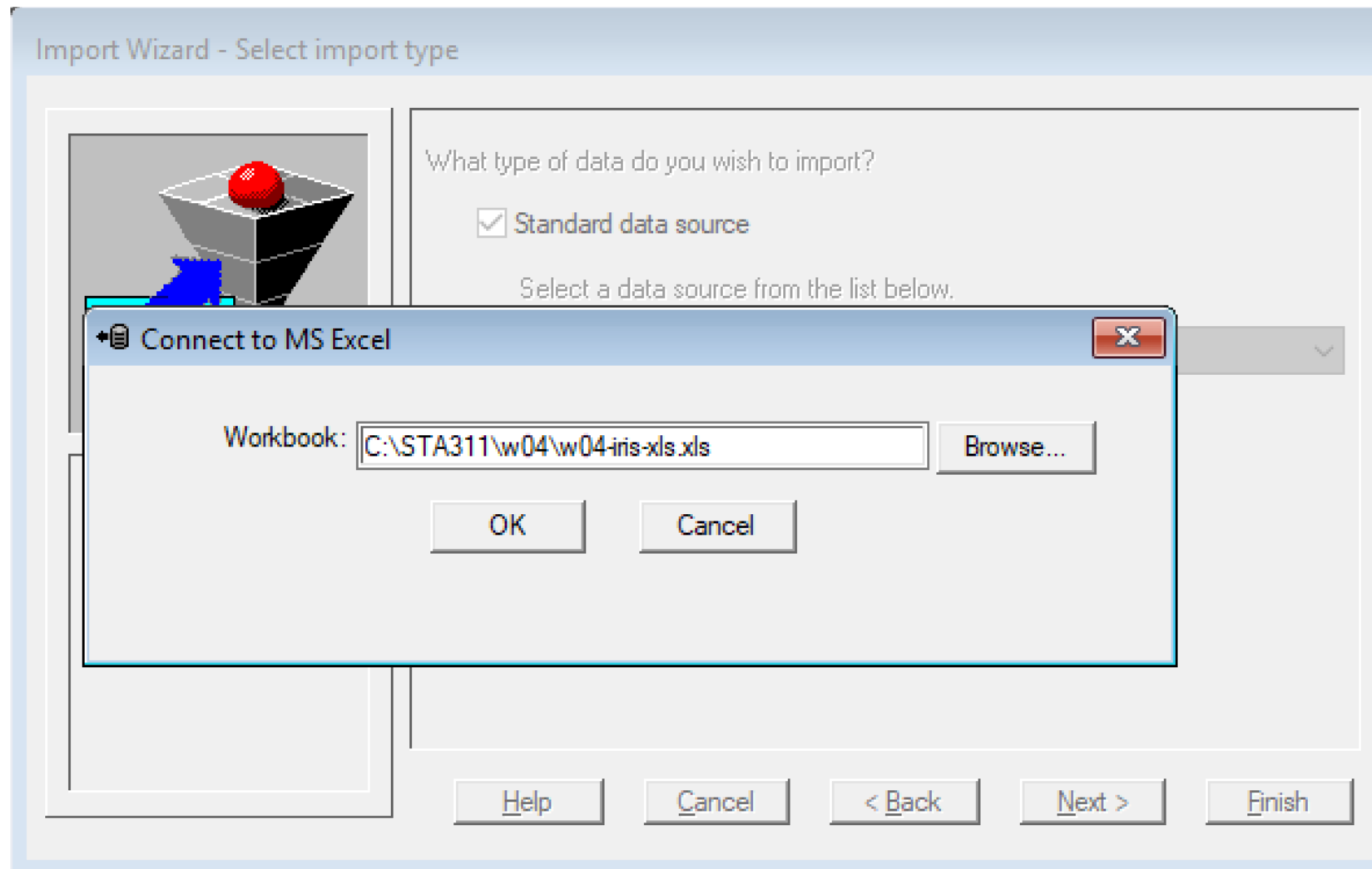
```
❏ DATA perm.myInDataset;  
    SET perm.MyExistingSASDataset;  
RUN;
```

### A Comparison between INPUT and SET

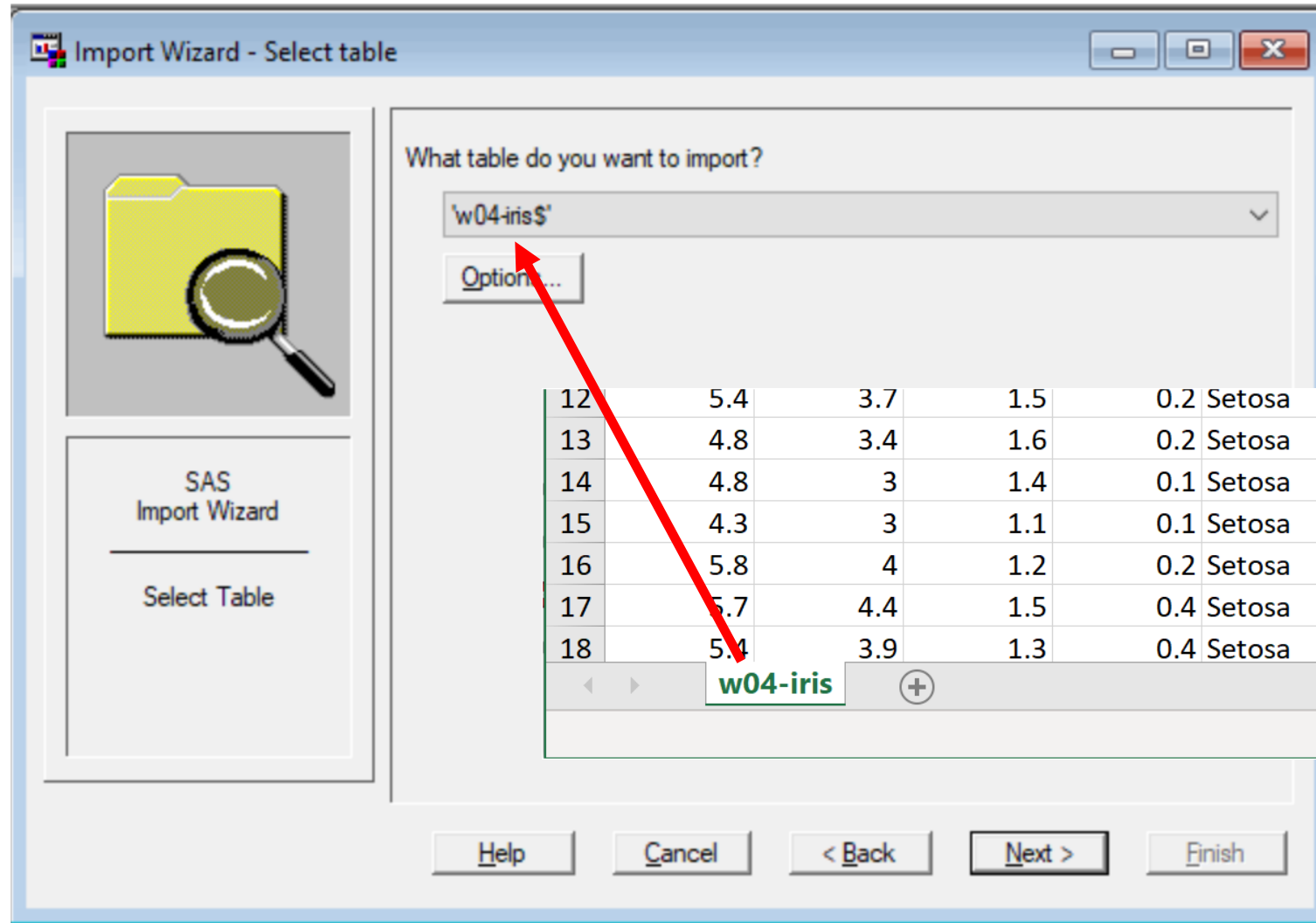
- ❏ SET reads an observation from an existing SAS data set.
- ❏ INPUT reads raw data from an external file (with INFILE statement) or from in-stream data lines (with DATALINES) in order to create SAS variables and observations.



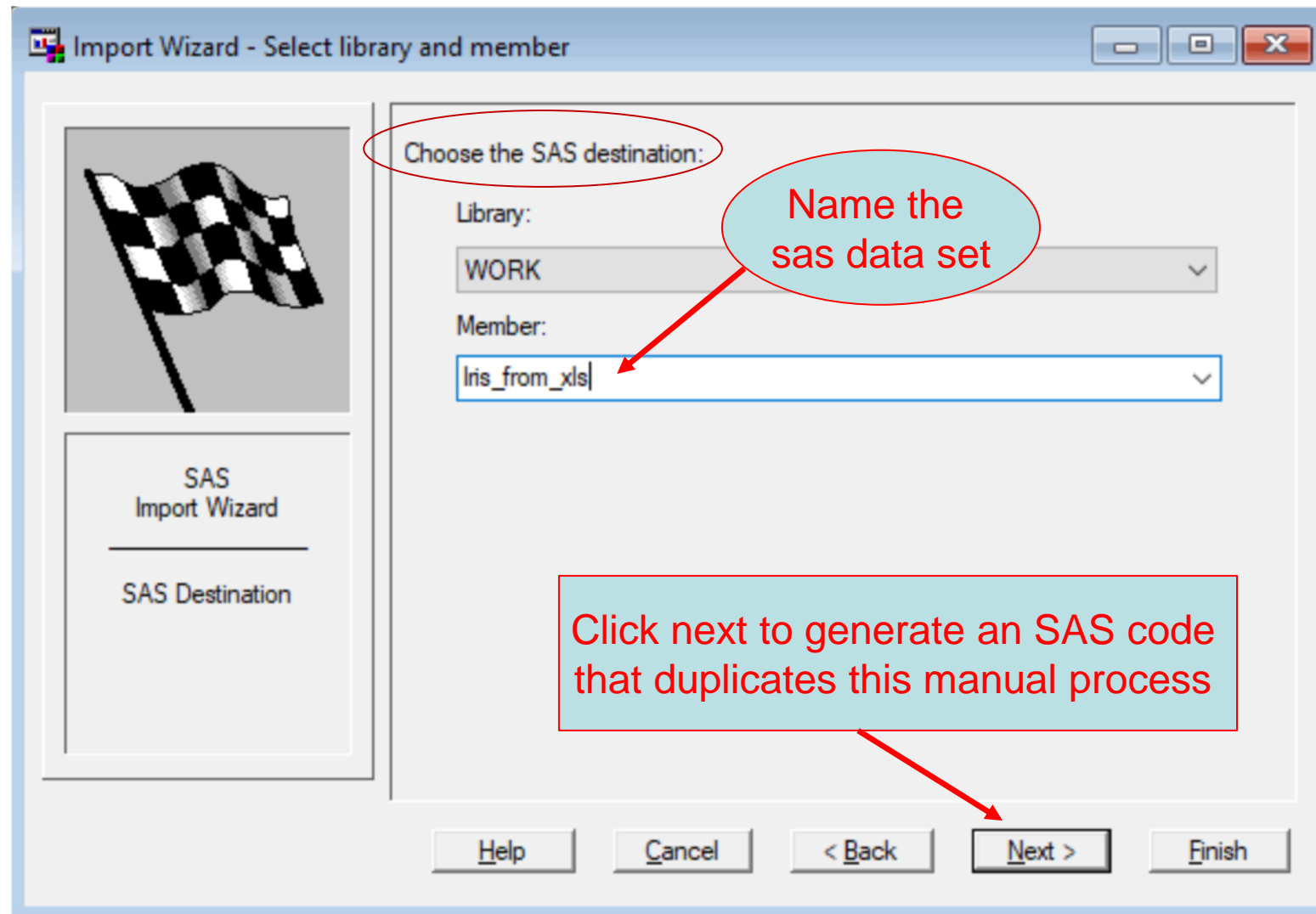
## Create SAS Data Sets from Other Applications: Locating the xls File



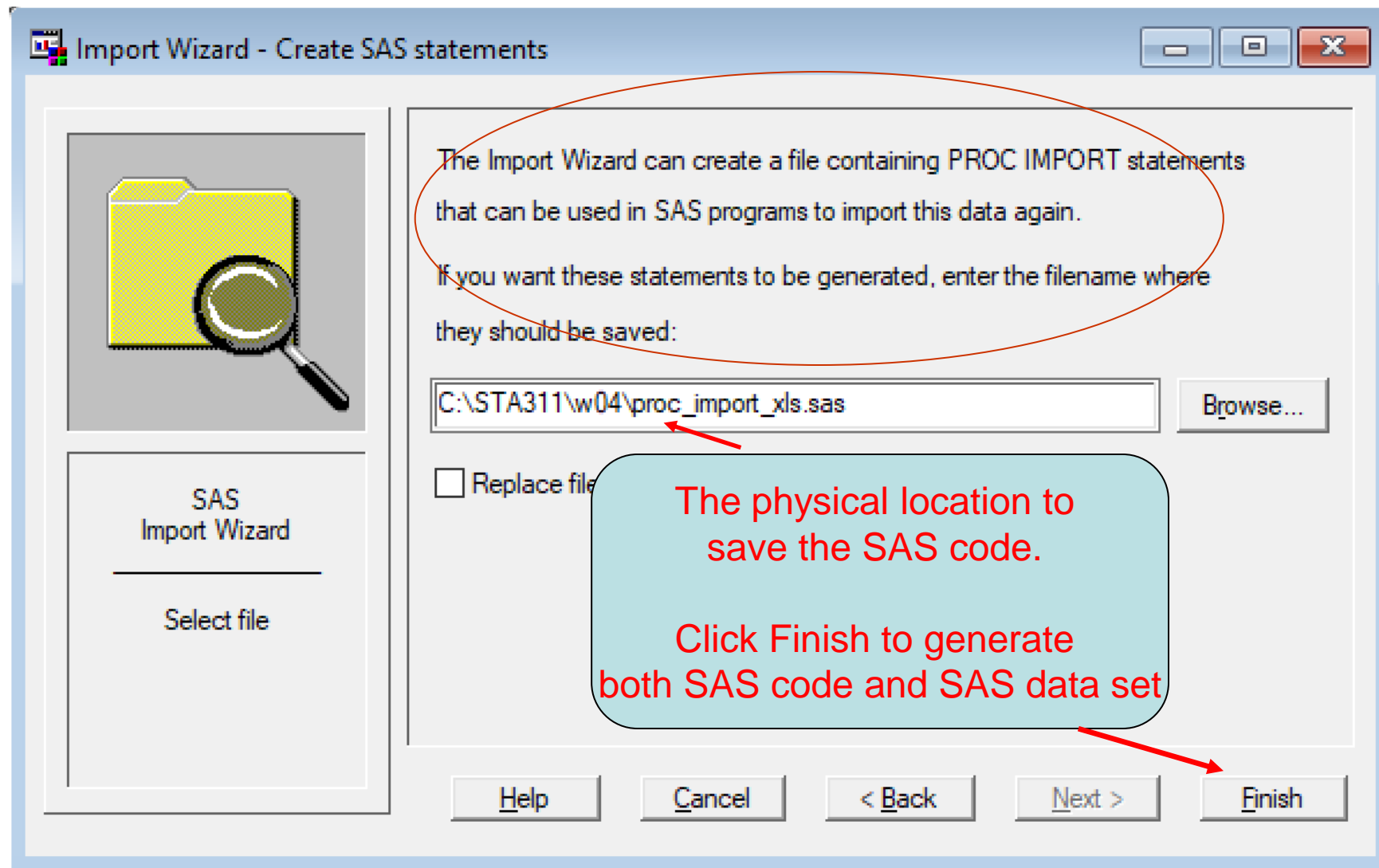
## Create SAS Data Sets from Other Applications: Specifying tab in xls File



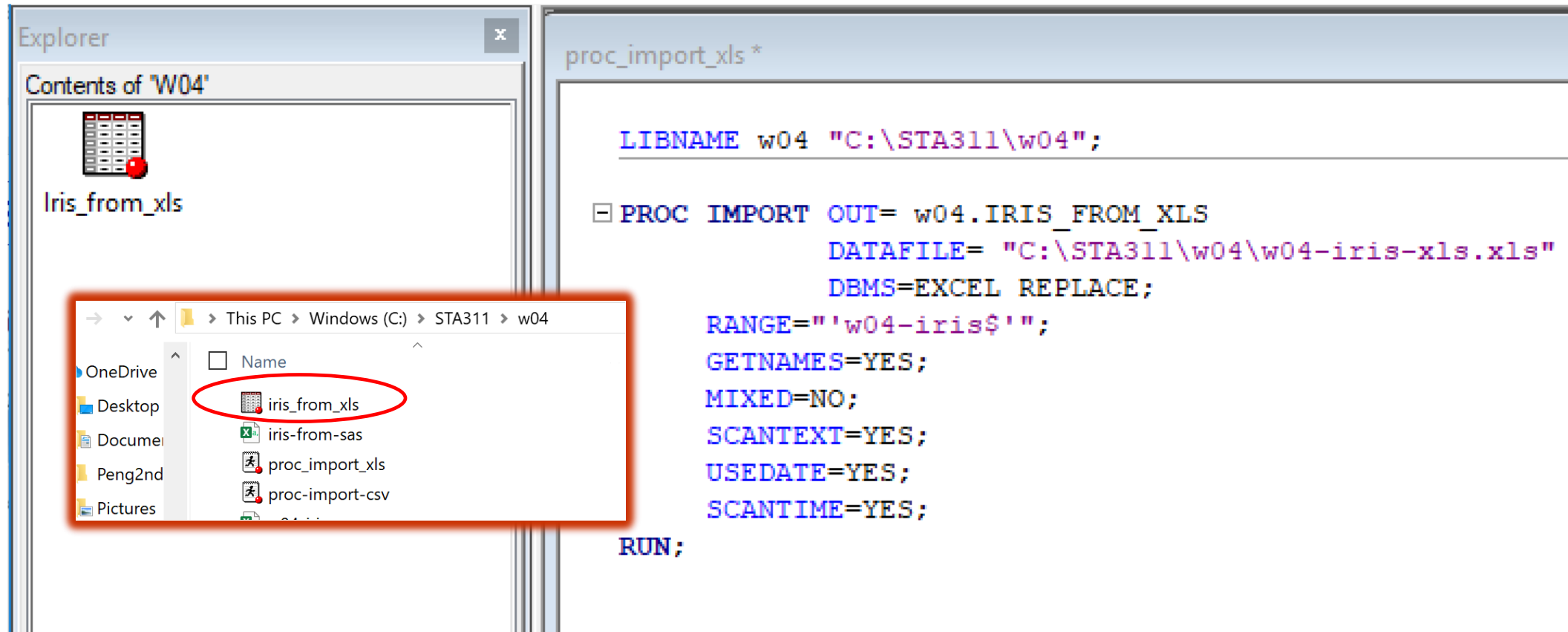
## Create SAS Data Sets from Other Applications: Save SAS Data Set



## Create SAS Data Sets from Other Applications: Save SAS Data Set



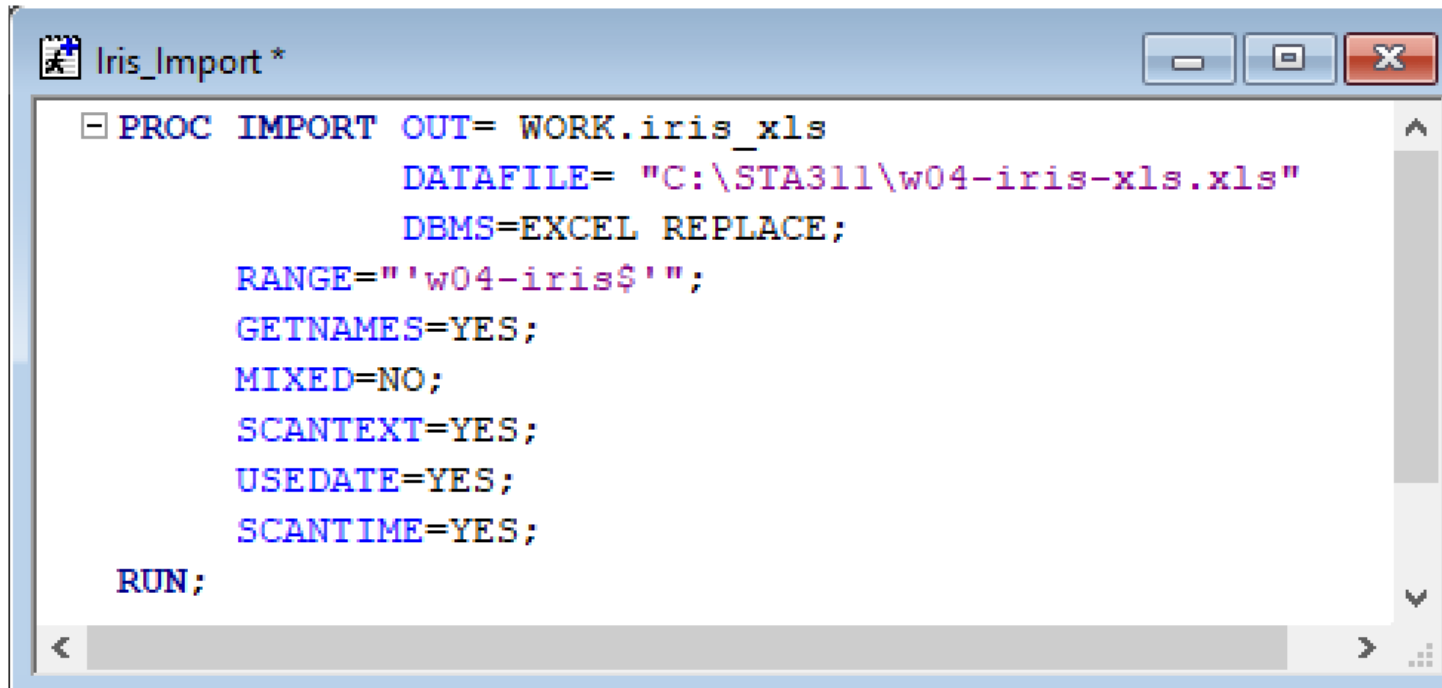
## Create SAS Data Sets from Other Applications: Save SAS Data Set



The screenshot displays the SAS interface. On the left, a file explorer window titled 'Contents of "W04"' shows the directory structure. The path is 'This PC > Windows (C:) > STA311 > w04'. The files listed are 'iris\_from\_xls' (highlighted with a red circle), 'iris-from-sas', 'proc\_import\_xls', and 'proc-import-csv'. On the right, the SAS code editor shows the following code:

```
proc_import_xls *  
  
LIBNAME w04 "C:\STA311\w04";  
  
PROC IMPORT OUT= w04.IRIS_FROM_XLS  
            DATAFILE= "C:\STA311\w04\w04-iris-xls.xls"  
            DBMS=EXCEL REPLACE;  
  
            RANGE="'w04-iris$';"  
            GETNAMES=YES;  
            MIXED=NO;  
            SCANTEXT=YES;  
            USEDATE=YES;  
            SCANTIME=YES;  
  
RUN;
```

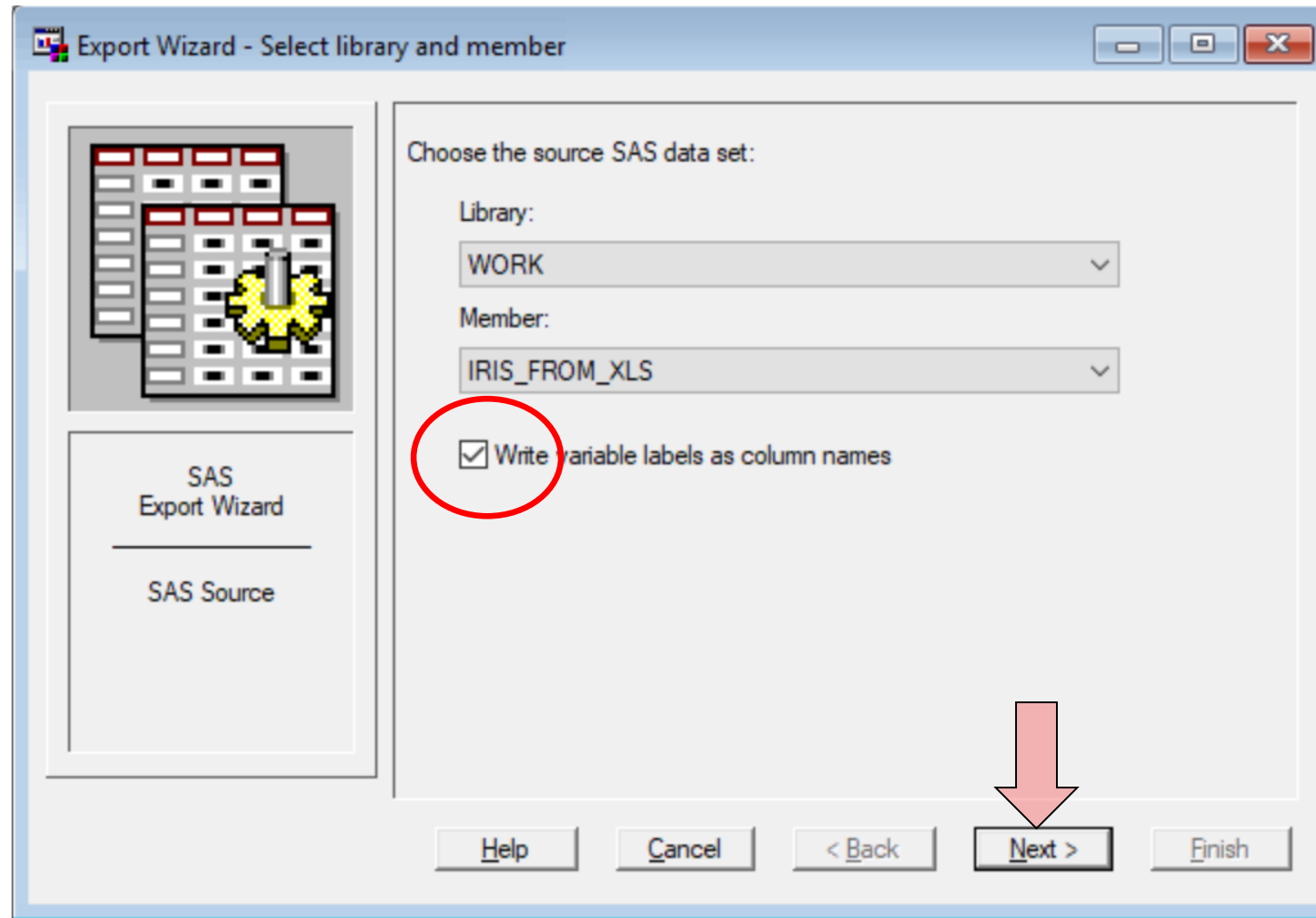
# SAS Generated PROC IMPORT Code



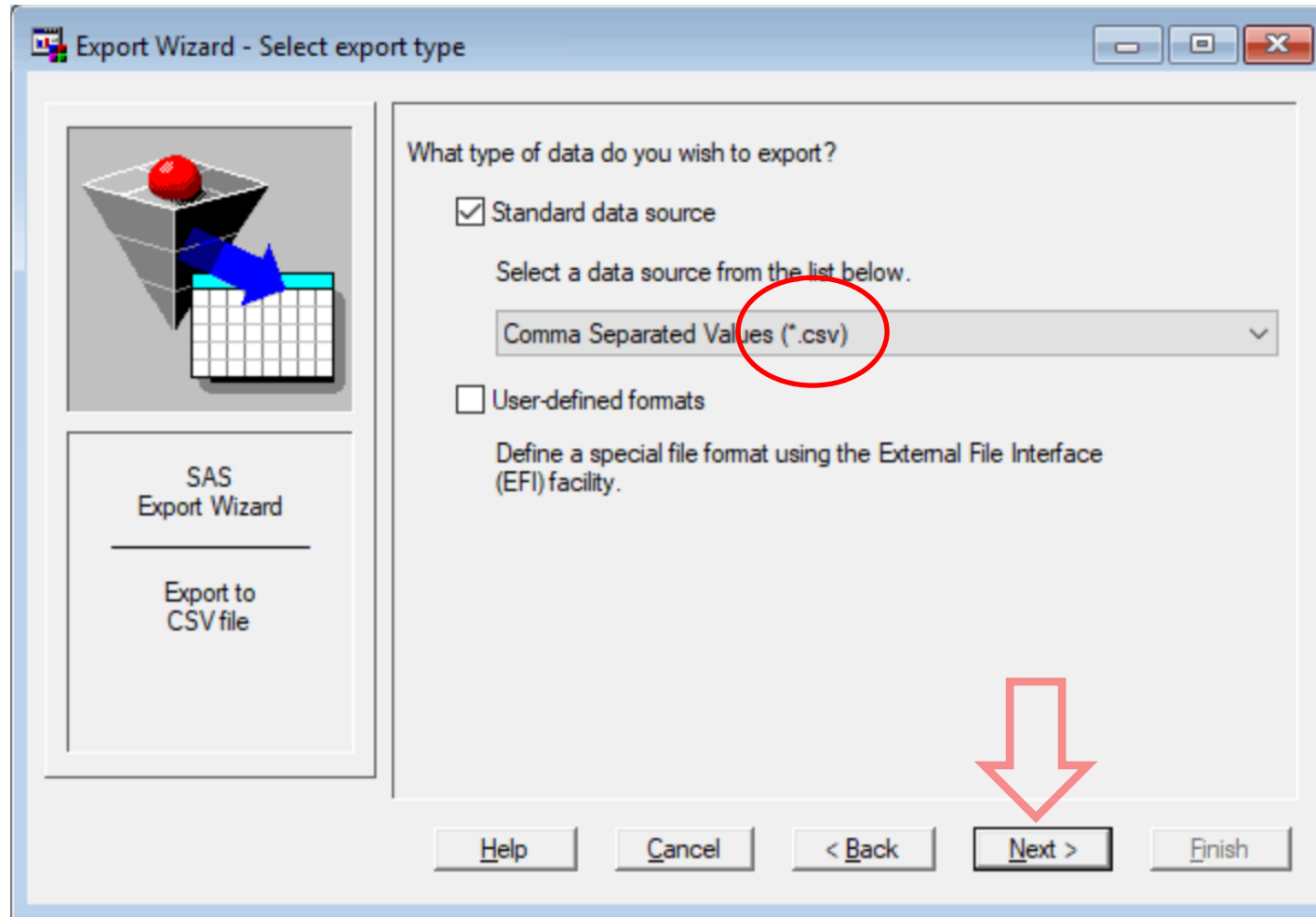
```
PROC IMPORT OUT= WORK.iris_xls
            DATAFILE= "C:\STA311\w04-iris-xls.xls"
            DBMS=EXCEL REPLACE;
            RANGE="'w04-iris$'";
            GETNAMES=YES;
            MIXED=NO;
            SCANTEXT=YES;
            USEDATE=YES;
            SCANTIME=YES;

RUN;
```

# Export SAS Data in CSV File

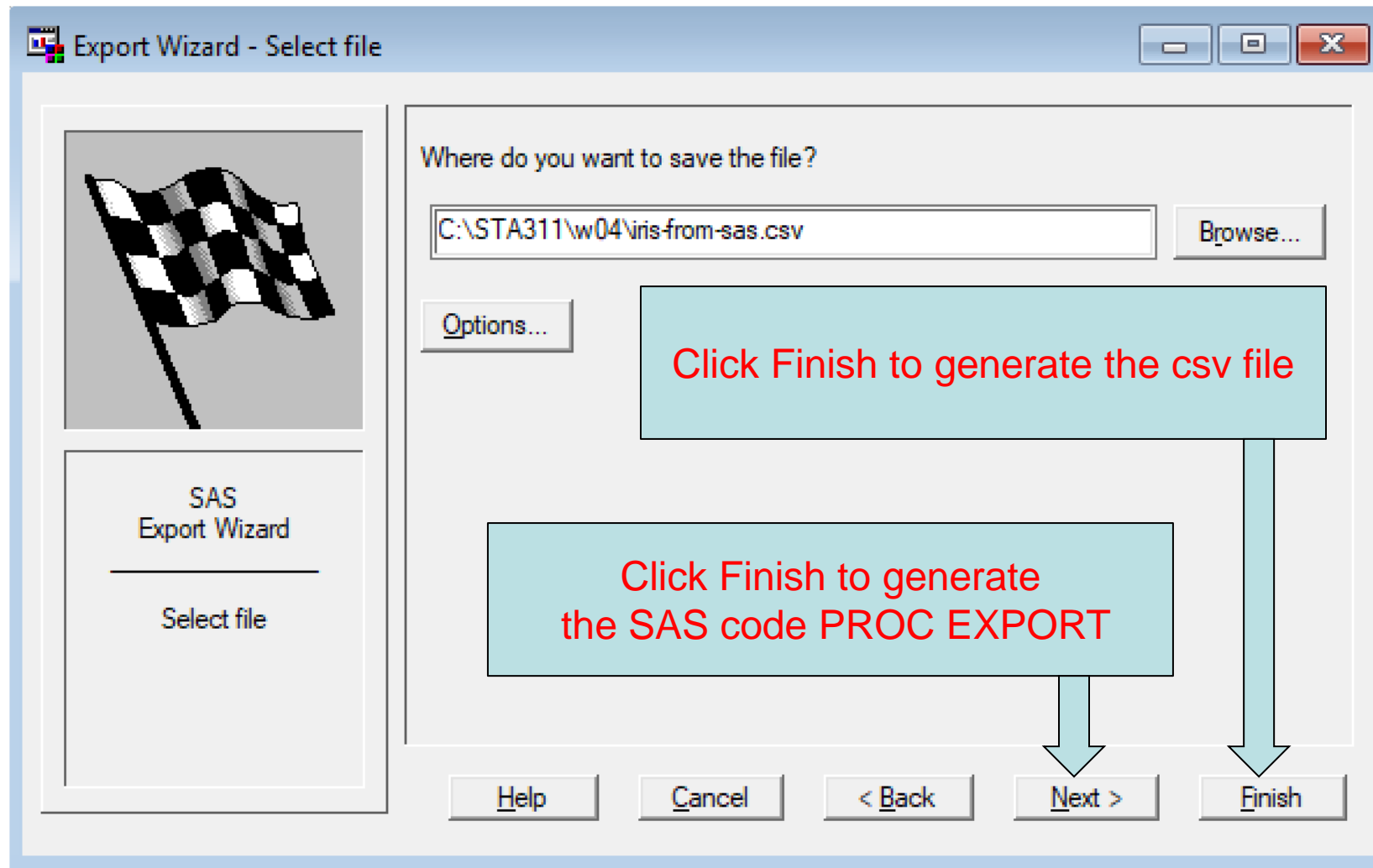


# Export SAS Data in CSV File





# Export SAS Data in CSV File



# SAS Generated PROC EXPORT

```
❏ PROC IMPORT OUT= w04.IRIS_FROM_XLS  
              DATAFILE= "C:\STA311\w04\w04-iris-xls.xls"  
              DBMS=EXCEL REPLACE;  
              RANGE="'w04-iris$';  
              GETNAMES=YES;  
              MIXED=NO;  
              SCANTEXT=YES;  
              USEDATE=YES;  
              SCANTIME=YES;  
RUN;
```

## Input Wizard v.s. PROC IMPORT/EXPORT

If you occasionally load a few data files, there is no difference between input Wizard and PROC IMPORT/EXPORT.

However,

If you work in a data-rich environment in industries such as financial services, healthcare, medicine, transportation, pharmaceutical companies, and some government sectors, you may have to deal with hundreds or thousands of files or streaming data sources. The input/output wizard approach is not a feasible choice. Using PROC IMPORT/EXPORT can automate your manual work!

## A Piece of Advice on Output Data

You may not have control over the format of input data.  
But you have the full control over the output data formats.

Unless you are asked to output data in a specific format,  
you should always export your SAS data set in CSV  
format.