UCD School of Mathematics and Statistics

STAT40840: Data programming with SAS
Laura Kirwan

Lecture 1

www.ucd.ie/mathstat

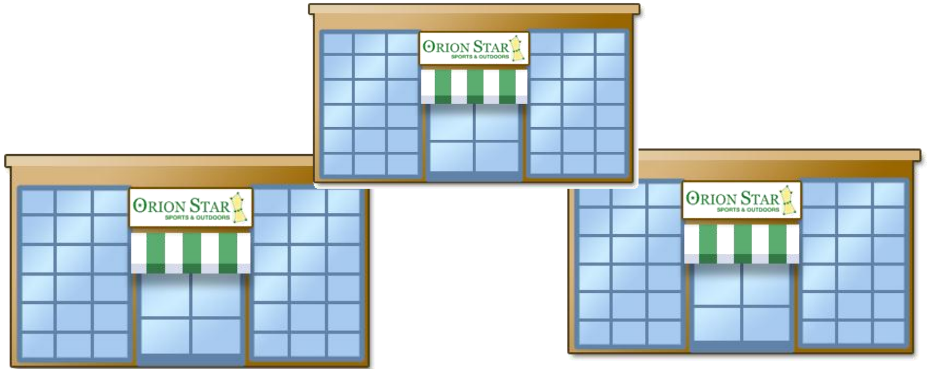
Objectives Week 1

1. Describe the data that is used in the course.
2. Specify the naming convention that is used for the course files.
3. Describe the SAS programming interfaces
4. Introduce SAS programs
5. Submitting a SAS program
6. Program syntax

Objective 1 – Course data

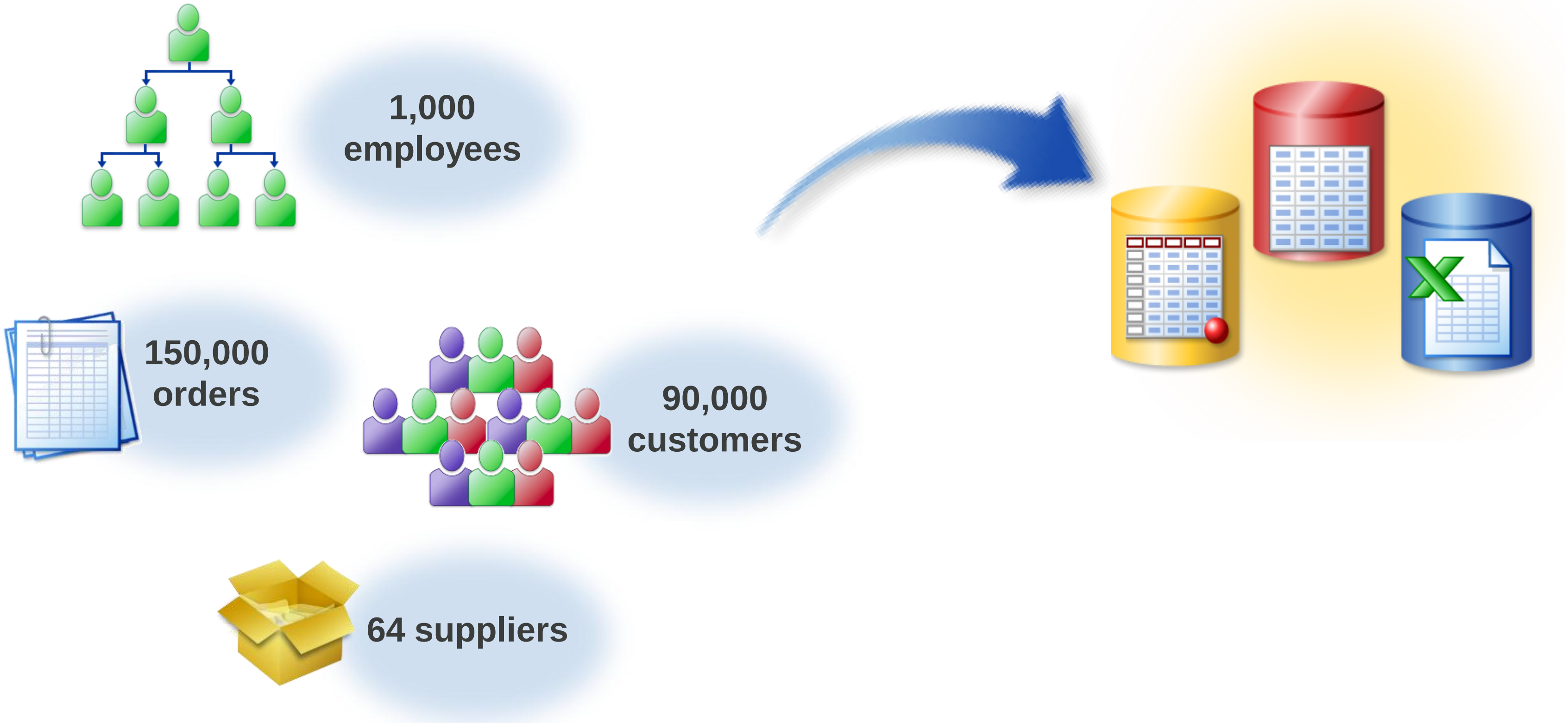
Orion Star Sports & Outdoors

This datasets for this course focus on a fictitious global sports and outdoors retailer that has traditional stores, an online store, and a catalog business.



Orion Star Data

- Large amounts of data are stored in various formats.



1,000 employees

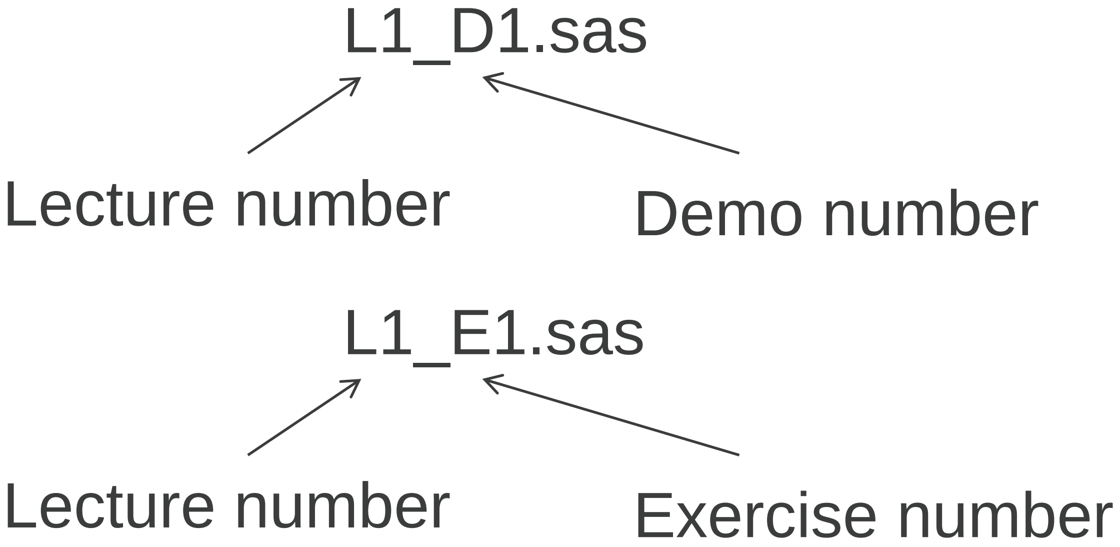
150,000 orders

90,000 customers

64 suppliers

Objective 2 – naming convention

- In this course, I will use the structure below to name SAS programs.



L1_D1.sas

Lecture number

Demo number

L1_E1.sas

Lecture number

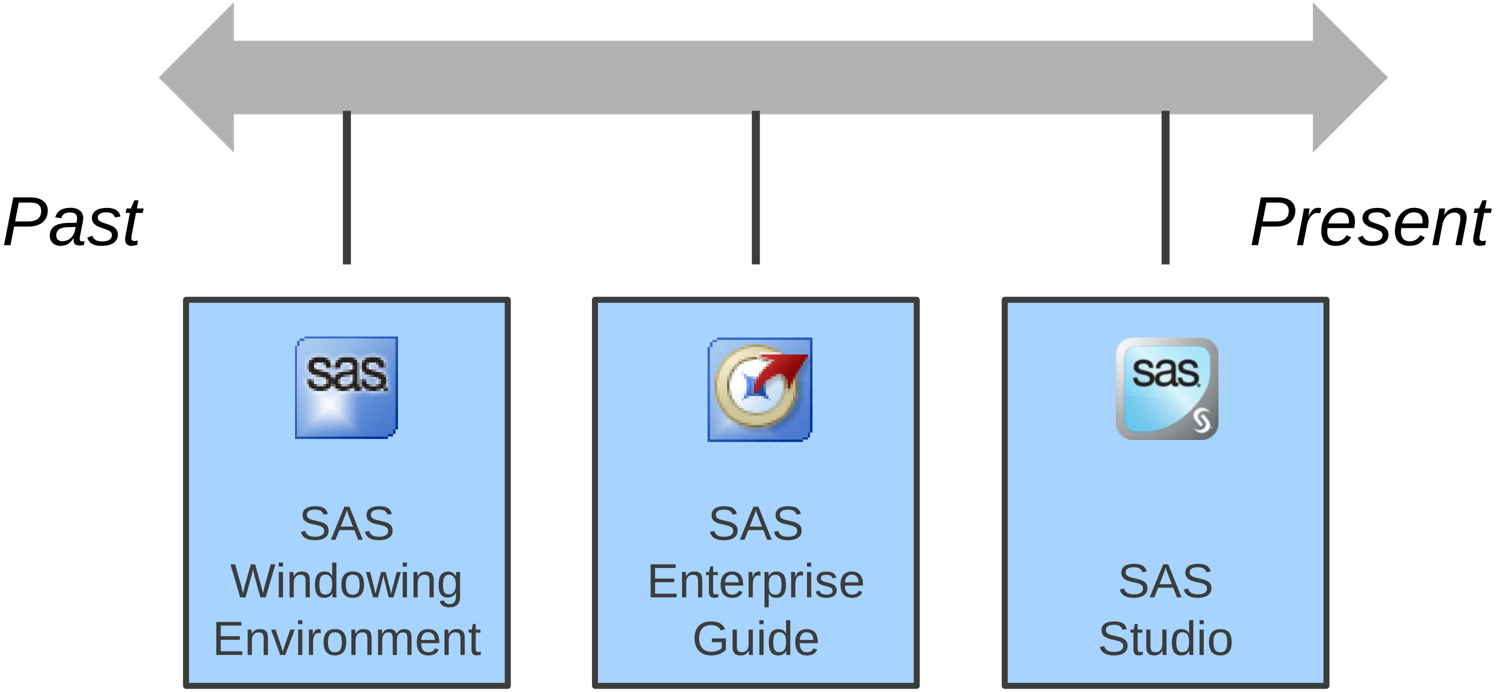
Exercise number

Objective 3 - SAS Interfaces

There are three possible SAS interfaces for processing a SAS program in interactive mode.

- SAS Windowing Environment
- SAS Enterprise Guide
- SAS Studio

SAS Interfaces



Past

Present

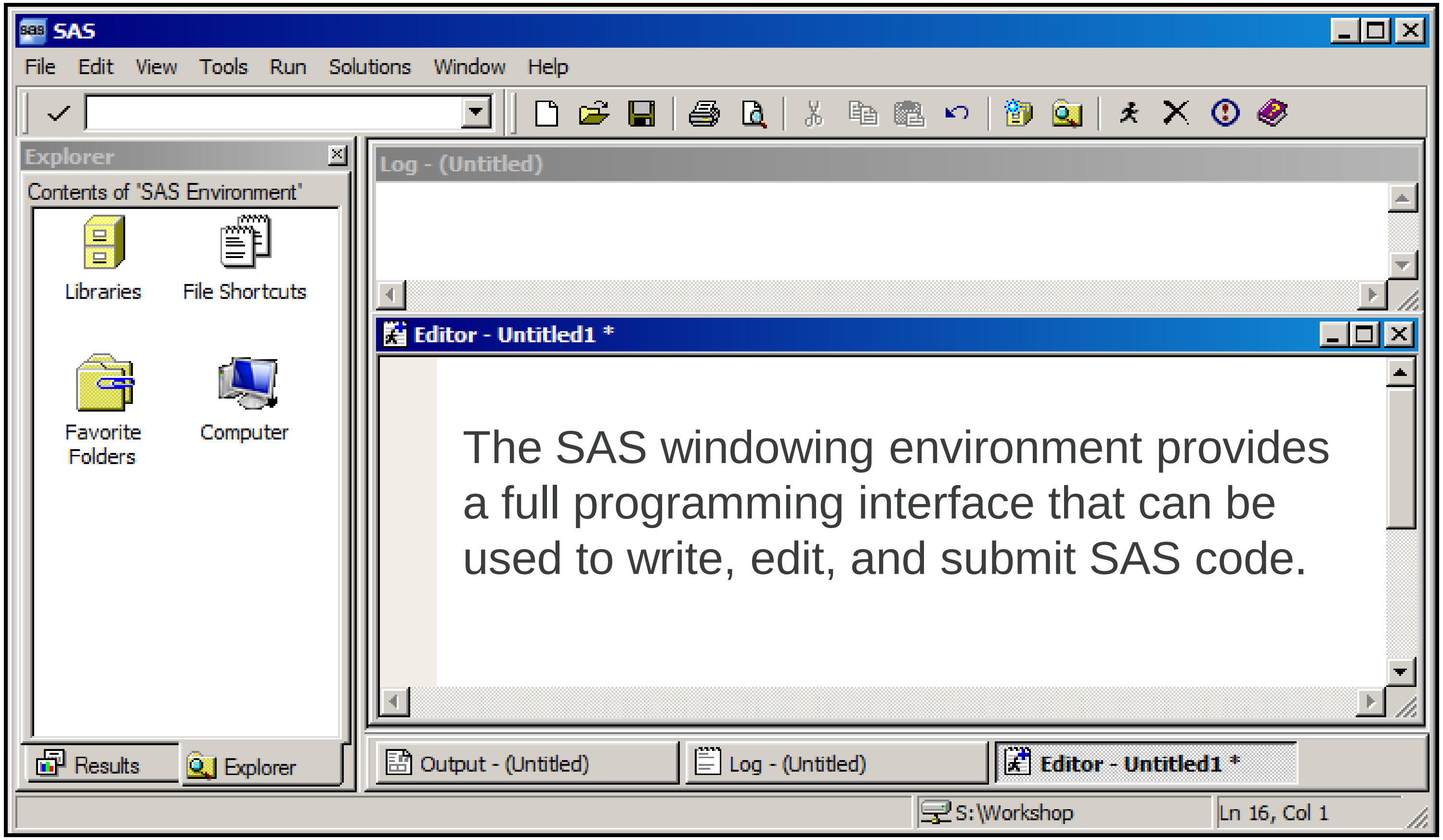
SAS Windowing Environment

SAS Enterprise Guide

SAS Studio

SAS Windowing Environment

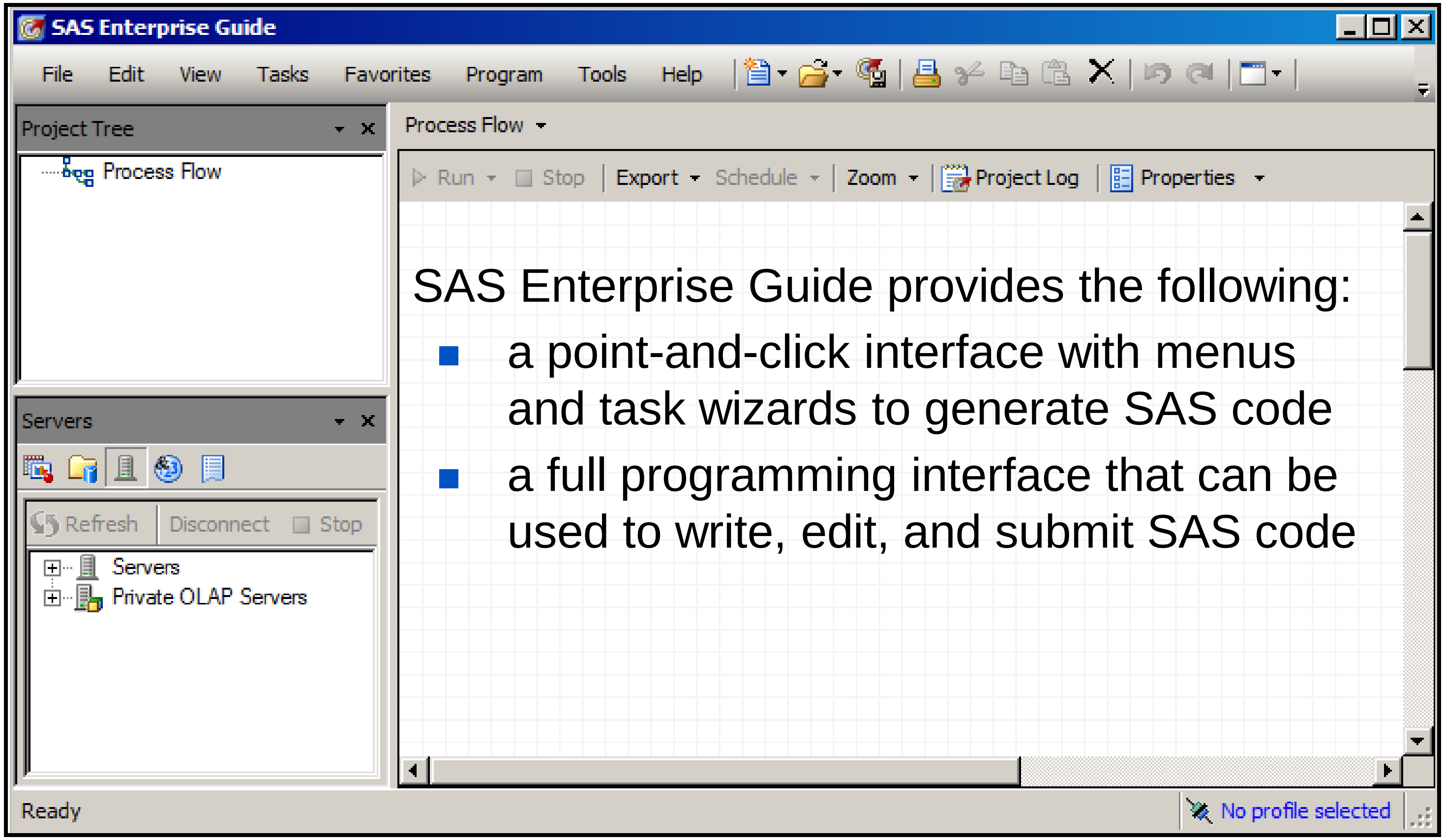
The *SAS windowing environment* is an application that is accessed from different operating environments.



The SAS windowing environment provides a full programming interface that can be used to write, edit, and submit SAS code.

SAS Enterprise Guide

SAS Enterprise Guide is a client application that is accessed from the Windows operating environment.

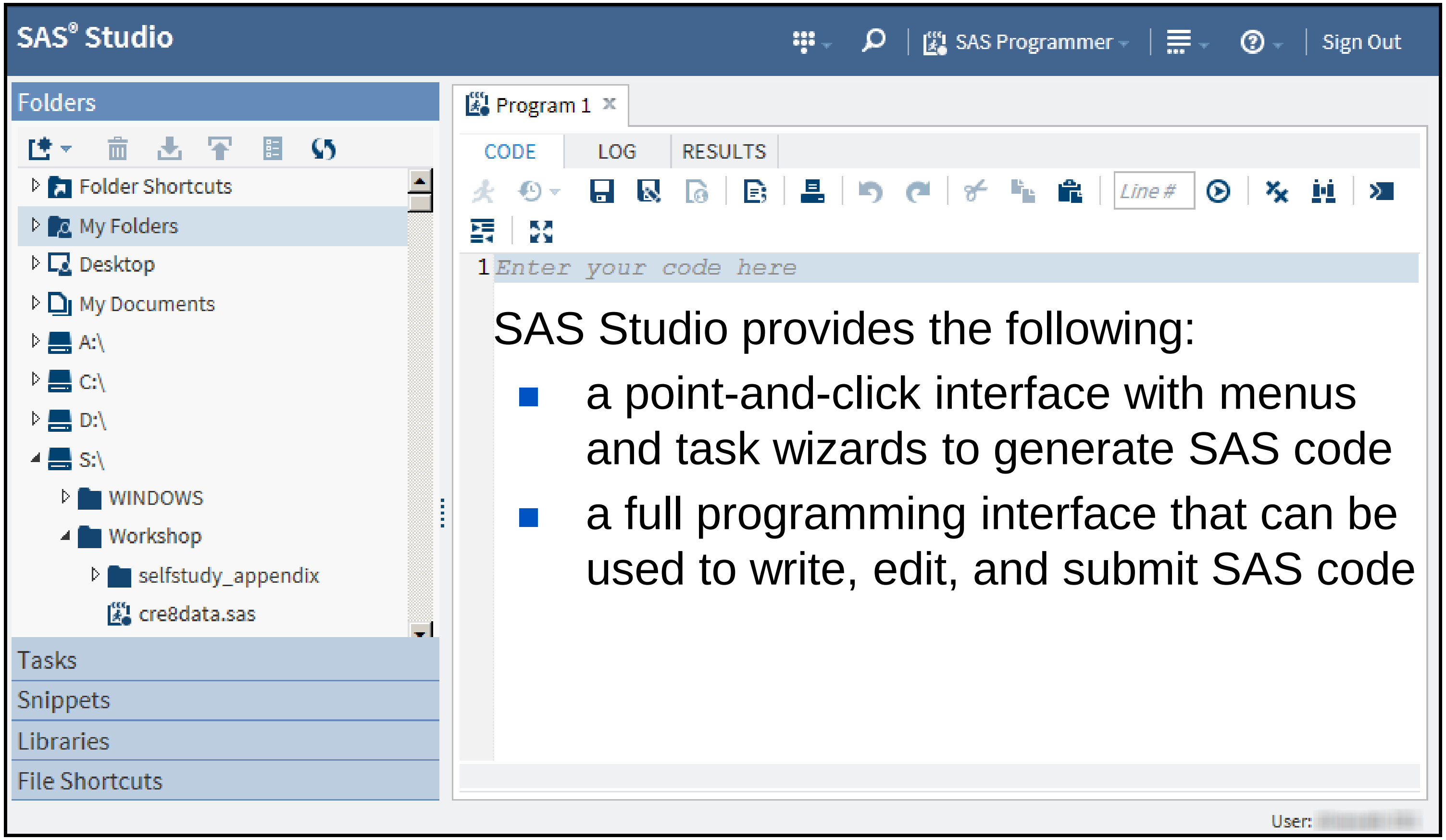


SAS Enterprise Guide provides the following:

- a point-and-click interface with menus and task wizards to generate SAS code
- a full programming interface that can be used to write, edit, and submit SAS code

SAS Studio

SAS Studio is a web client that is accessed through an HTML5-compliant web browser.



SAS Studio provides the following:

- a point-and-click interface with menus and task wizards to generate SAS code
- a full programming interface that can be used to write, edit, and submit SAS code

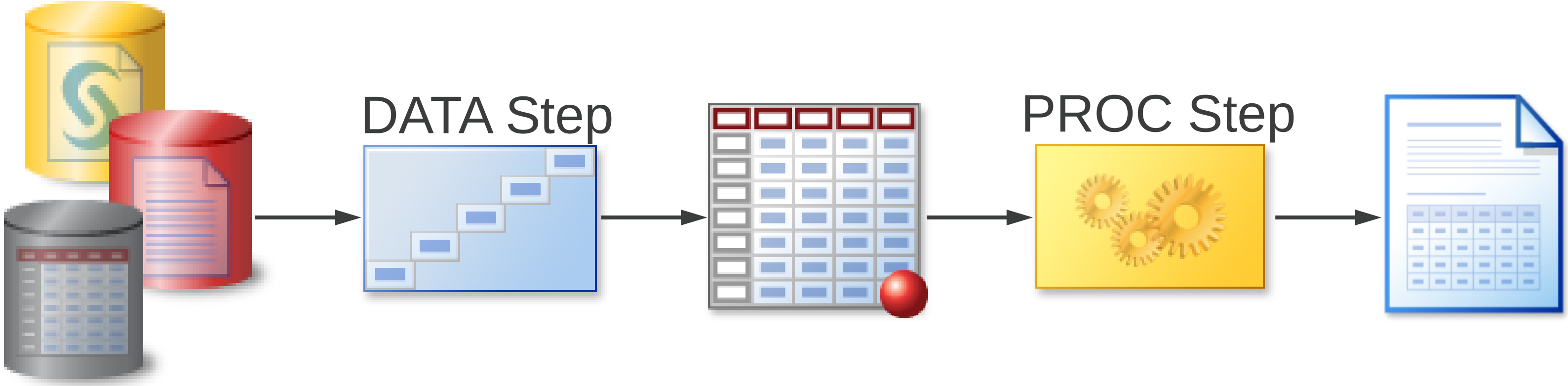
SAS Interface Tabs or Windows

Regardless of the SAS interface that you choose to use, there are three primary tabs or windows.

Editor	Enter, edit, submit, and save a SAS program
Log	Browse notes, warnings, and errors relating to a submitted SAS program
Results	Browse output from reporting procedures

Objective 4 - SAS Programs

- A *SAS program* is a sequence of one or more steps.



– *DATA steps* typically create SAS data sets.

– *PROC steps* typically process SAS data sets to generate reports and graphs, and to manage data.

SAS Program Steps

- A *step* is a sequence of SAS statements. This program has a DATA step and a PROC step.

```
data work.newemps;
   infile "&path\newemps.csv" dlm=',';
   input First $ Last $ Title $ Salary;
run;

proc print data=work.newemps;
run;
```

Step Boundaries

- SAS steps begin with either of the following:
  - a DATA statement
  - a PROC statement

- SAS detects the end of a step when it encounters one of the following:
  - a RUN statement (for most steps)
  - a QUIT statement (for some procedures)
  - the beginning of another step (DATA statement or PROC statement)

Steps

- How many steps are in program **L1_D1**?

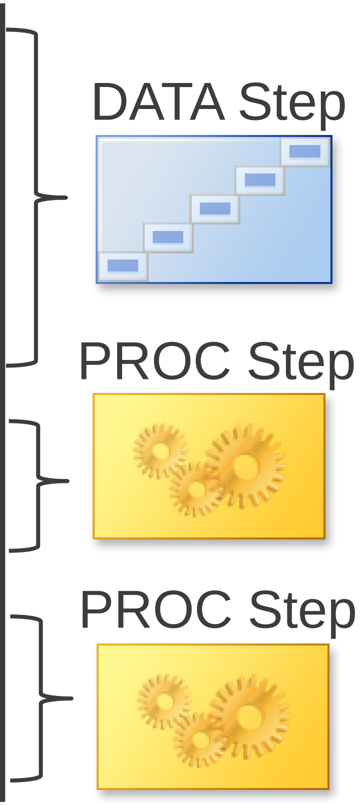
```
data work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.newsalesemps;
run;

proc means data=work.newsalesemps;
   var Salary;
run;
```

L1_D1.sas

Steps

- How many steps are in program **L1_D1**? **three**

```
data work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.newsalesemps;
run;

proc means data=work.newsalesemps;
   var Salary;
run;
```



SAS Program Example

This DATA step creates a temporary SAS data set named **work.newsalesemps** by reading four fields from a file.

```
data work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.newsalesemps;
run;

proc means data=work.newsalesemps;
   var Salary;
run;
```

L1_D1.sas

SAS Program Example

This PROC PRINT step lists the **work.newsalesemps** data set.

```
data work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.newsalesemps;
run;

proc means data=work.newsalesemps;
   var Salary;
run;
```

L1_D1.sas

SAS Program Example

This PROC MEANS step summarizes the **Salary** variable in the **work.newsalesemps** data set.

```
data work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.newsalesemps;
run;

proc means data=work.newsalesemps;
   var Salary;
run;
```

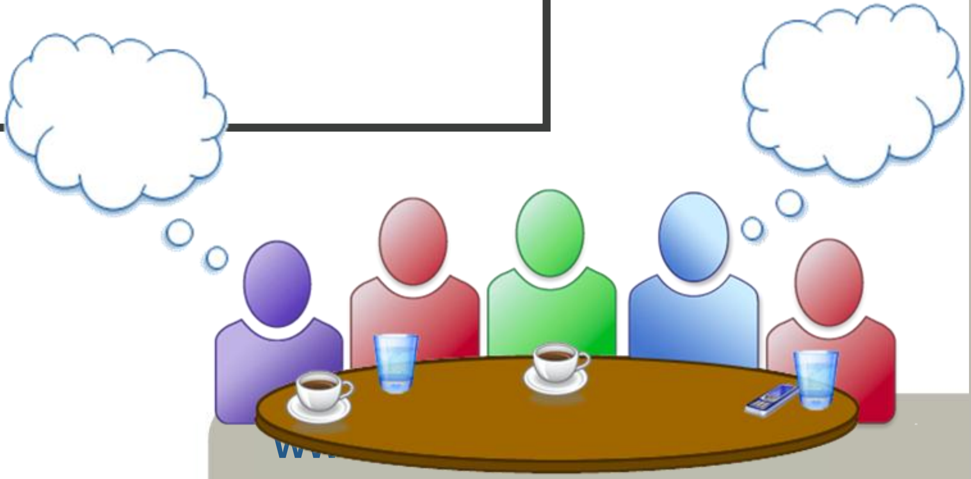
L1_D1.sas

Exercise 1

How does SAS detect the end of each step in this program?

```
data work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.newsalesemps;

proc means data=work.newsalesemps;
   var Salary;
quit;
```

L1_E1.sas



Processing Modes

The following are two possible processing modes for submitting a SAS program:

Interactive Mode	A SAS program is submitted within a SAS interface for foreground processing
Batch Mode	A SAS program is submitted to the operating environment for background processing

✏ In this course, interactive mode is used to process SAS programs.

Objective 5

Submitting SAS Programs in the SAS Windowing Environment

⇨ Demonstration illustrating how to use the Editor, Log, and Results Viewer windows within the SAS windowing environment.

Activities

SAS Windowing Environment	1. Creating Course Data in the SAS Windowing Environment 2. Exploring the SAS Windowing Environment 3. Accessing Help and Documentation

Objective 6 - Program Syntax

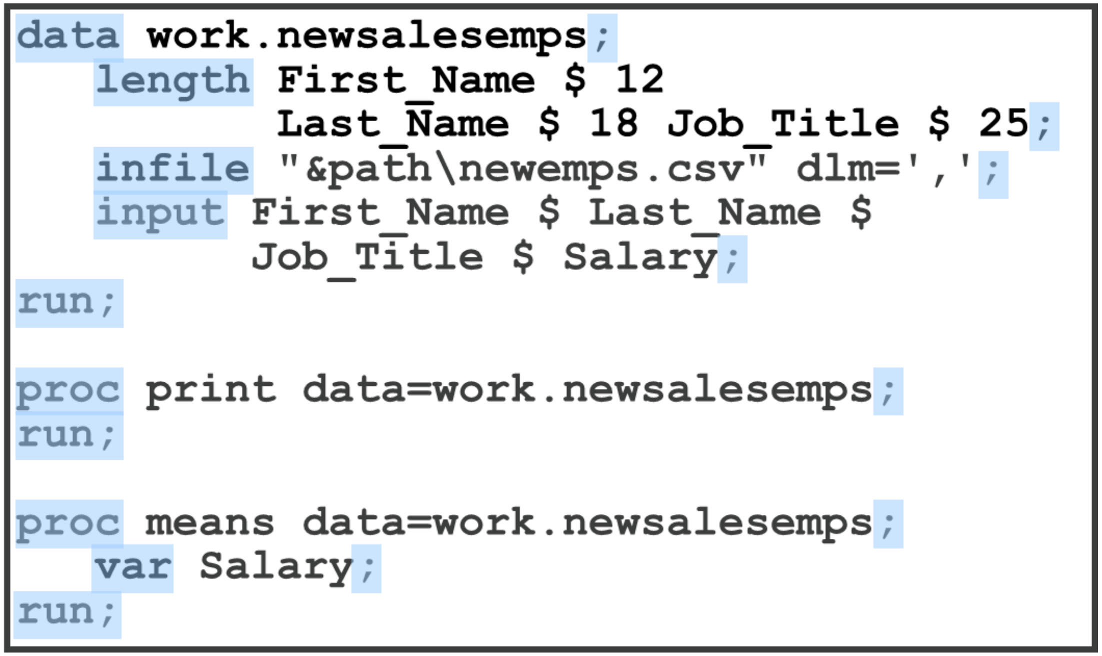
- Identify the characteristics of SAS statements.
- Define SAS syntax rules.
- Document a program using comments.
- Diagnose and correct a program with errors.

SAS programs

Well-formatted, clearly documented SAS programs are an industry best practice.

SAS Syntax Rules: Statements

- *SAS statements*
  - usually begin with an ***identifying keyword***
  - always end with a ***semicolon***.



```
data work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.newsalesemps;
run;

proc means data=work.newsalesemps;
   var Salary;
run;
```

L1_D1.sas

Exercise 2

- How many statements make up this DATA step?

L1_E2.sas

```
data work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;
```

SAS Program Structure

- SAS code is free format.

```
data work.newsalesemps;
length First_Name $ 12
Last_Name $ 18 Job_Title $ 25;
infile "&path\newemps.csv" dlm=',';
input First_Name $ Last_Name $
Job_Title $ Salary;run;
proc print data=work.newsalesemps; run;
   proc means data  =work.newsalesemps;
var Salary;run;
```

L1_D2.sas

- This program is syntactically correct but difficult to read.

SAS Program Structure

- Rules for SAS Statements
  - Statements can begin and end in any column.
  - A single statement can span multiple lines.
  - Several statements can appear on the same line.
  - Unquoted values can be lowercase, uppercase, or mixed case.

```
data work.newsalesEmps;
length First_Name $ 12
Last_Name $ 18 Job_Title $ 25;
infile "&path\newemps.csv" dlm=',';
input First_Name $ Last_Name $
Job_Title $ Salary;run;
proc print data=work.newsalesemps; run;
   proc means data  =work.newsalesemps;
var Salary;run;
```

unconventional formatting

Recommended Formatting

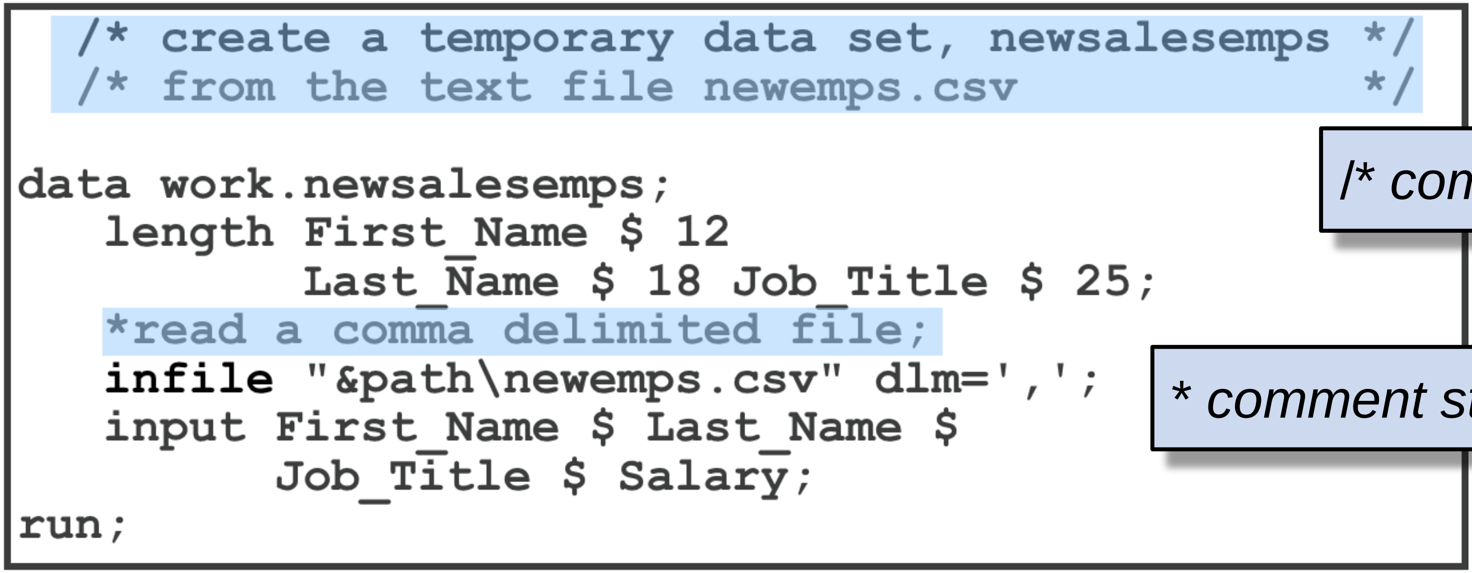
- Begin each statement on a new line.
- Use white space to separate words and steps.
- Indent statements within a step.
- Indent continued lines in multi-line statements.

```
data work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.newsalesemps;
run;

proc means data=work.newsalesemps;
   var Salary;
run;
```

conventional formatting

Program Documentation

You can embed comments in a program as explanatory text.



```
  /* create a temporary data set, newsalesemps */
  /* from the text file newemps.csv            */

data work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   *read a comma delimited file;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;
```

/ comment */*

** comment statement ;*

SAS ignores comments during processing but writes them to the SAS log.

SAS Comments

This program contains four comments.

```
*----------------------------------------*
|   This program creates and uses the    |
|   data set called work.newsalesemps.   |   ❶
*----------------------------------------*;
data work.newsalesemps;
   length First_Name $ 12 Last_Name $ 18
          Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary /* numeric */;❷
run;
  /*
proc print data=work.newsalesemps;   ❸
run;
  */
proc means data=work.newsalesemps;
   * var Salary ;❹
run;
```

L1_D3.sas

Exercise 3

- Open and examine **L1_E3.sas**. Based on the comments, which steps do you think are executed and what output is generated?

- Submit the program. Which steps were executed?

Syntax Errors

A *syntax error* is an error in the spelling or grammar of a SAS statement. SAS finds syntax errors as it compiles each SAS statement, before execution begins.

- Examples of syntax errors:
  - misspelled keywords
  - unmatched quotation marks
  - missing semicolons
  - invalid options

Exercise 4

This program includes three syntax errors. One is an invalid option. What are the other two syntax errors?

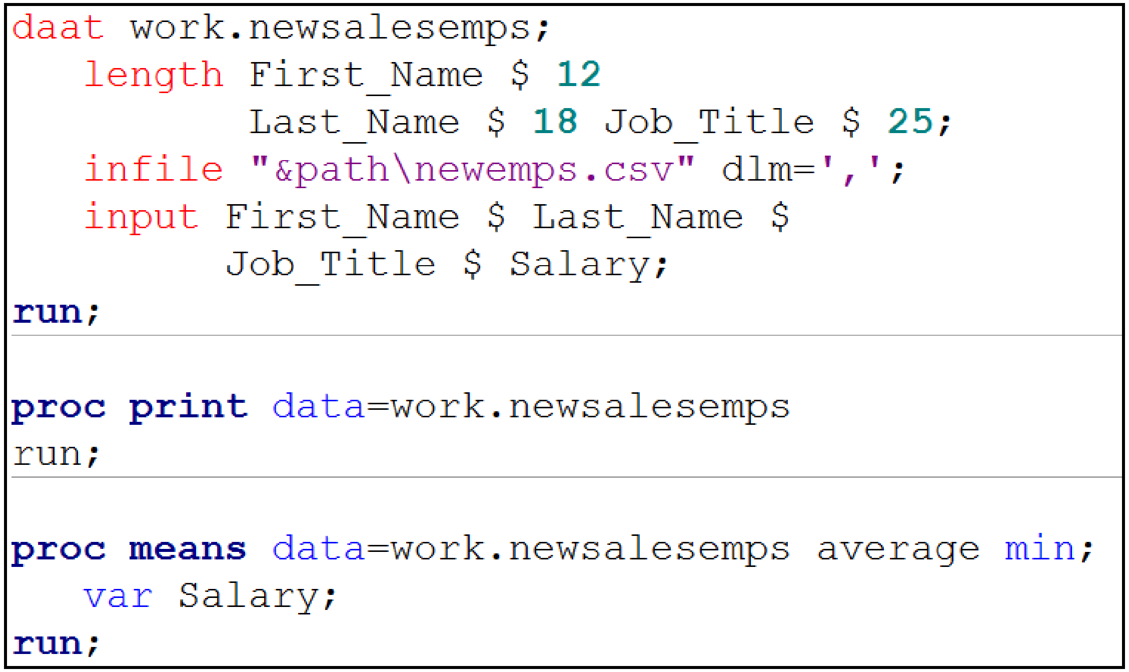
```
daat work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.newsalesemps
run;

proc means data=work.newsalesemps average min;
   var Salary;
run;
```

L1_E4.sas

invalid option

Syntax Errors

Syntax errors in a SAS program can possibly be detected based on the color of the syntax on the Editor tab or in the window.



```
daat work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',';
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.newsalesemps
run;

proc means data=work.newsalesemps average min;
   var Salary;
run;
```

L1_D4.sas

Syntax Errors

When SAS encounters a syntax error, it writes a warning or error message to the log.

```
WARNING 14-169: Assuming the symbol DATA was misspelled as daat.
```

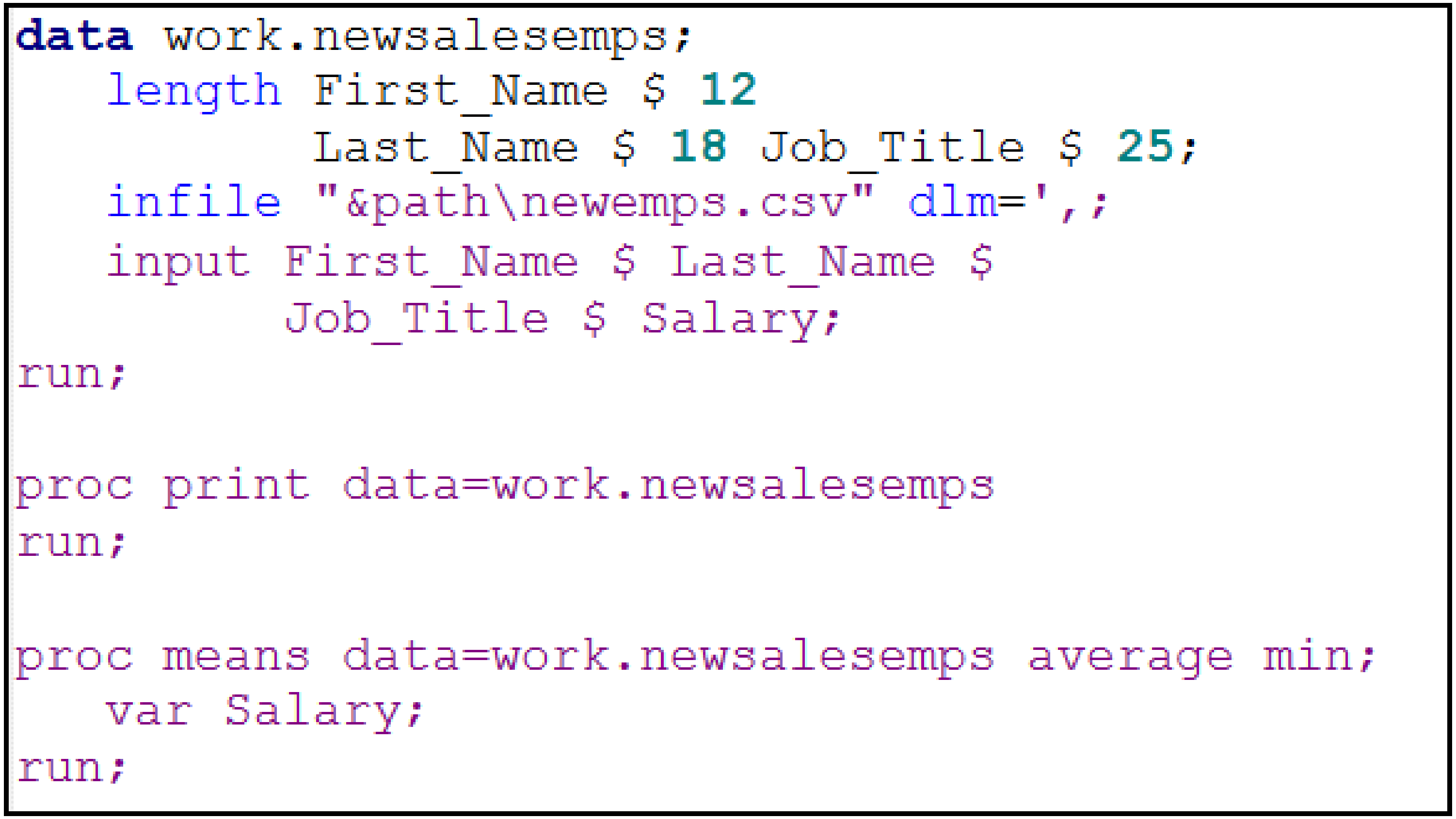
```
ERROR 22-322: Syntax error, expecting one of the following: ;, (,
              BLANKLINE, CONTENTS, DATA, DOUBLE, GRANDTOTAL_LABEL,
              GRANDTOT_LABEL, GRAND_LABEL, GTOTAL_LABEL,
              GTOT_LABEL, HEADING, LABEL, N, NOOBS, NOSUMLABEL,
              OBS, ROUND, ROWS, SPLIT, STYLE, SUMLABEL, UNIFORM,
              WIDTH.
```

⚠ You should always check the log to make sure that the program ran successfully, even if output is generated.

Exercise 5

What is the syntax error in this program?

L1_E5.sas



```
data work.newsalesemps;
   length First_Name $ 12
          Last_Name $ 18 Job_Title $ 25;
   infile "&path\newemps.csv" dlm=',;
   input First_Name $ Last_Name $
         Job_Title $ Salary;
run;

proc print data=work.newsalesemps
run;

proc means data=work.newsalesemps average min;
   var Salary;
run;
```

Activities

SAS Windowing Environment	4. Correcting Quotation Marks in the SAS Windowing Environment 5. Diagnosing Errors in the SAS Windowing Environment