



Nov 2023

Master in Data Science and Business Analytics

SAS Laboratory Course  
Part 1



# Chapter 1: Introduction

**1.1 Overview**

**1.2 Introduction to programming**

**1.3 The Programming Process**

**1.4 Course Logistics**

# Chapter 1: Introduction

**1.1 Overview**

**1.2 Introduction to programming**

**1.3 The Programming Process**

**1.4 Course Logistics**

# What Is SAS?

SAS is a suite of business solutions and technologies to help organizations solve business problems.



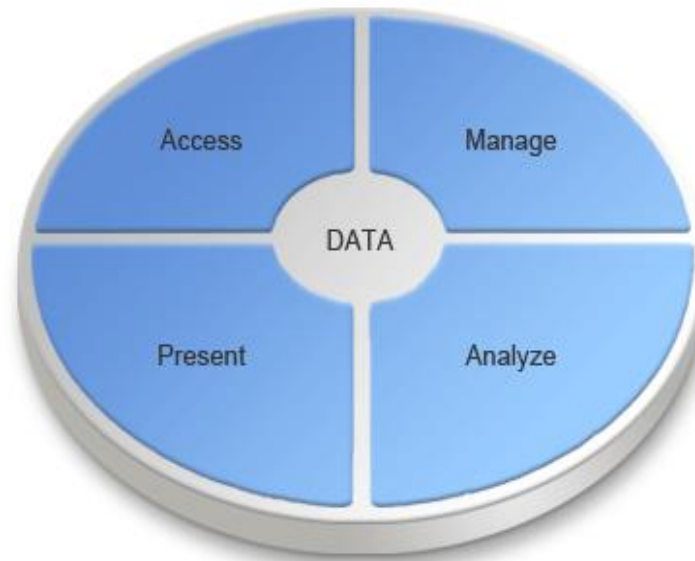
## History

SAS (pronounced "sass") once stood for "**statistical analysis system.**" It began at North Carolina State University as a project to analyze agricultural research. Demand for such software capabilities began to grow, and SAS was founded in 1976 to help customers in all sorts of industries – from pharmaceutical companies and banks to academic and governmental entities.

# Why Use SAS?

SAS enables you to do the following:

- access and manage data across multiple sources
- perform analyses and deliver information across your organization



Base SAS is the primary focus of this course.

# Chapter 1: Introduction

**1.1 Overview**

**1.2 Introduction to programming**

**1.3 The Programming Process**

**1.4 Course Logistics**

# Objectives

- Explain the need for computer programming.
- Identify the differences between using a programming language in a programming environment and in a point-and-click environment.

# Course Focus

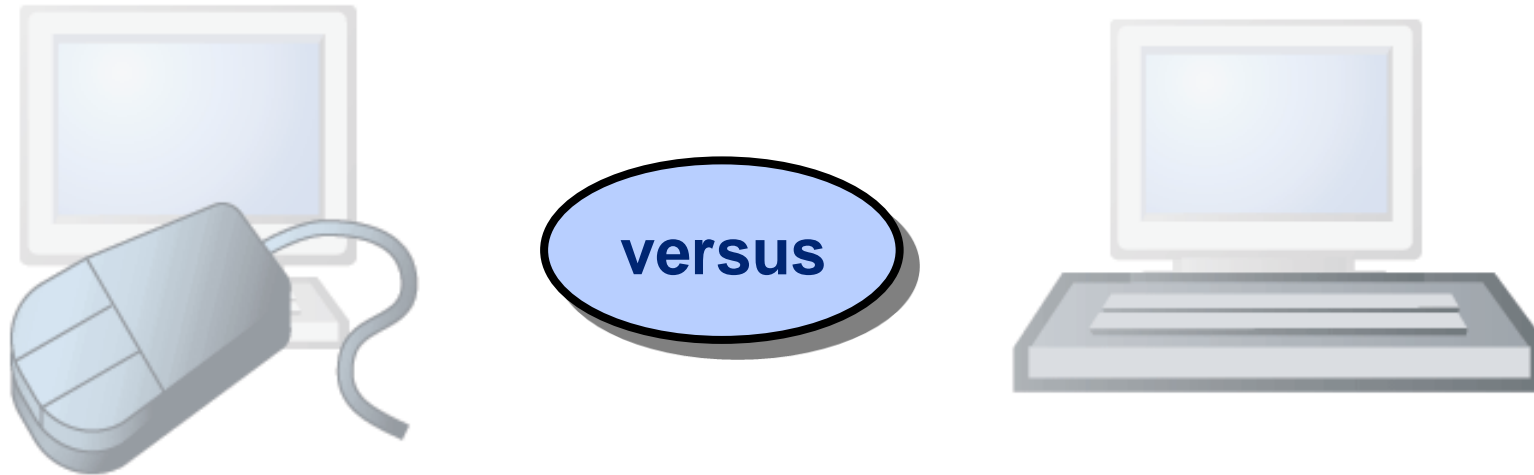
This course focuses on basic ***programming*** using SAS software.





# Point and Click versus Programming

Some computer products enable you to point and click or program using a programming language.



✍ SAS has both options, but in this class, you ***program*** with SAS.

# Point and Click

Pros	Cons
Intuitive to use	Cannot save choices
No syntax to learn	Limitations to capabilities
	Potentially limited to one operating environment

# Programming

Pros	Cons
Can save program	Not as intuitive to use
Broader capabilities	Must learn syntax
Potentially available on multiple operating environments	

## 1.01 Multiple Choice Poll

Please rate your computer programming experience.

- a. I have no programming experience.
- b. I have not written a program, but I run other people's programs.
- c. I have written a few simple programs.
- d. I have written a complex computer program.

# What Is Base SAS?

Base SAS is the primary focus of this course.

*Base SAS* is the foundation for all SAS software.

Base SAS provides the following:

- a highly flexible, highly extensible, fourth-generation programming language
- a rich library of encapsulated programming procedures
- a choice of programming interfaces



## 1.01 Multiple Choice Poll

What is your programming experience with SAS?

- a. maintaining programs written by others
- b. writing new programs
- c. no experience
- d. other

# Chapter 1: Introduction

**1.1 Overview**

**1.2 Introduction to programming**

**1.3 The Programming Process**

**1.4 Course Logistics**

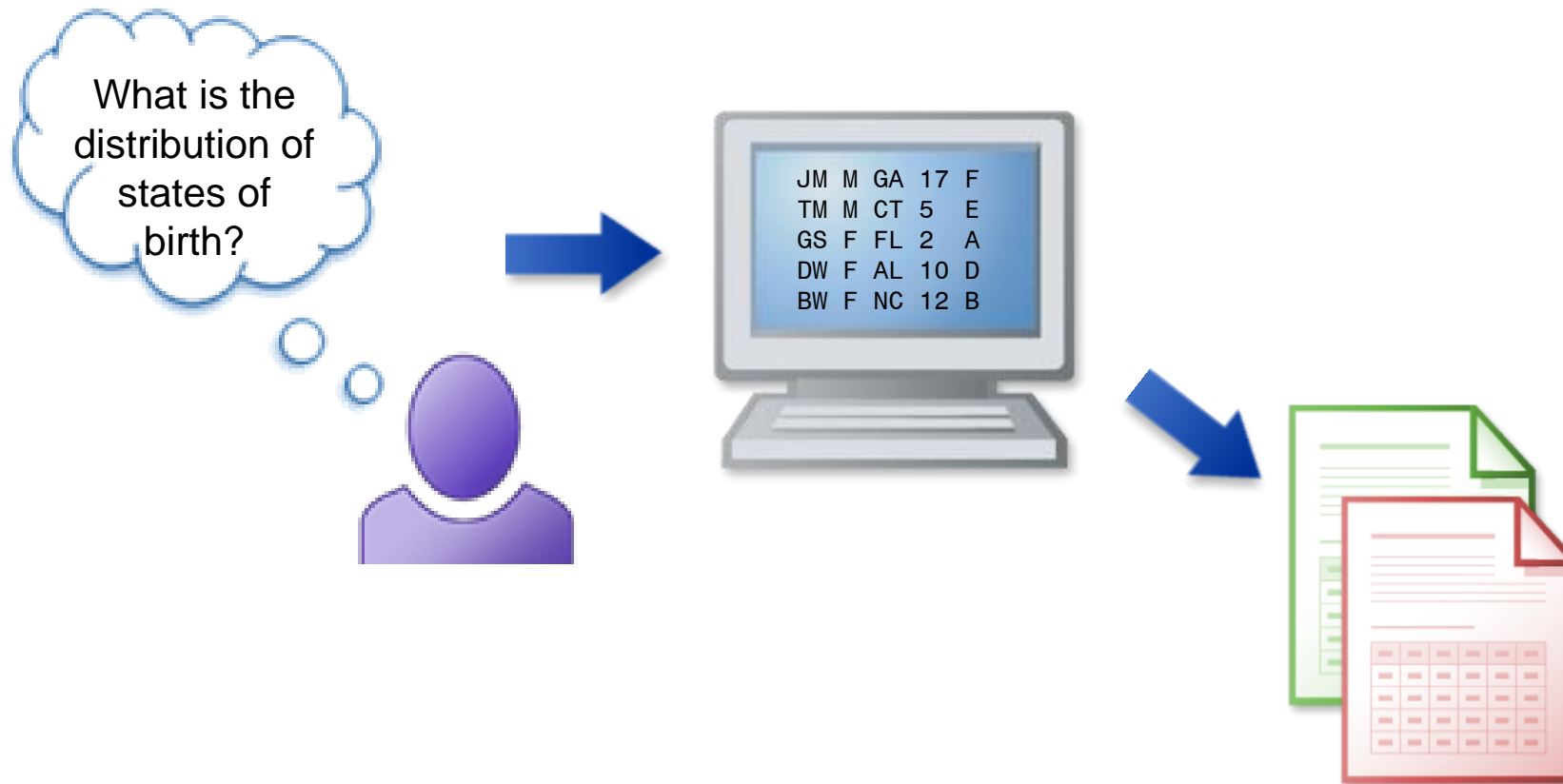
# Objectives

- Define the steps in the programming process.



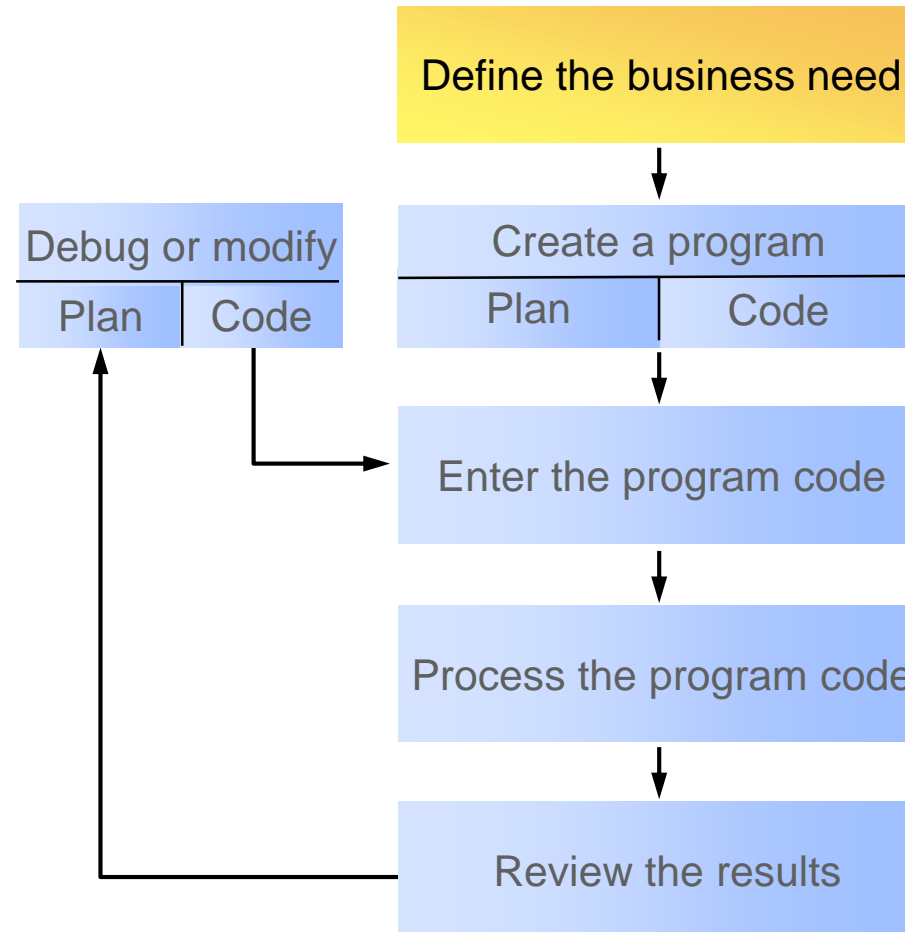
# The Programming Process

A programming process is needed to generate reports from any kind of data.



# The Programming Process

**Step 1:** Define the business need.

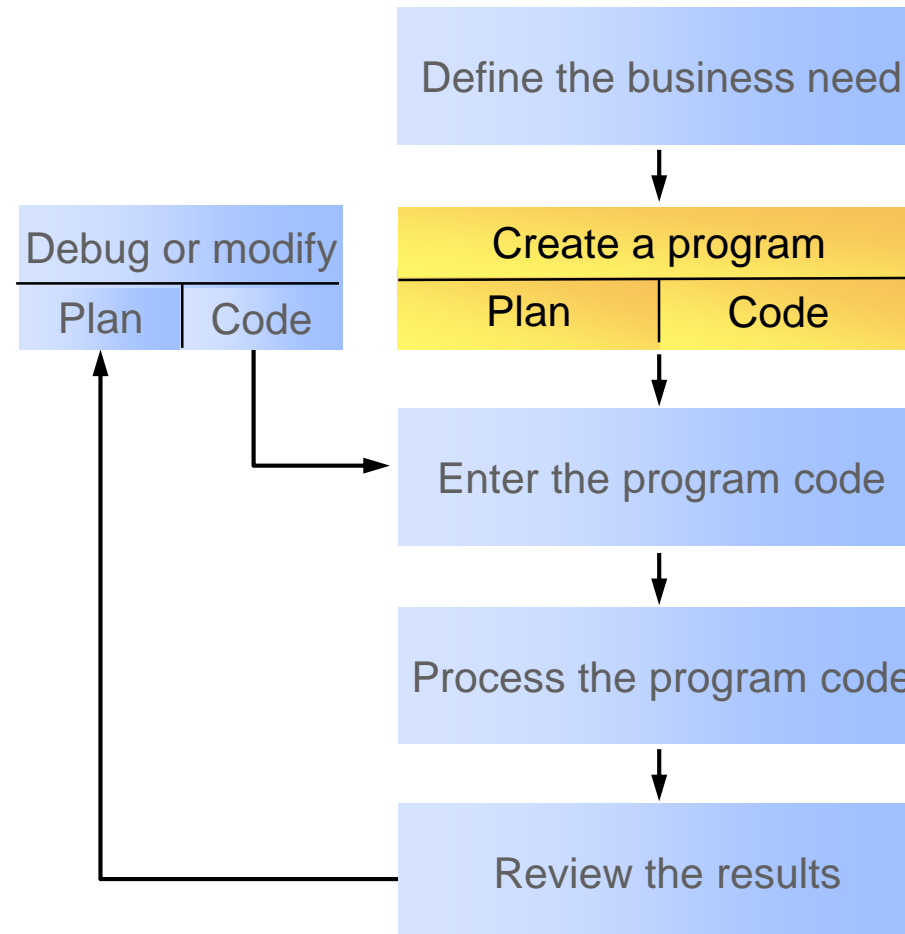


# The Programming Process

## **Step 2:** Create a program.

This step consists of two phases:

- planning the program
- coding the program

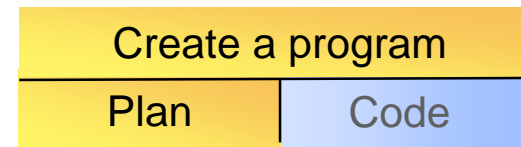


# The Programming Process

**Step 2:** Create a program (plan).

In the planning phase, you address these questions:

- What reports are desired?  
(*output*)
- What data will be used?  
(*input*)
- How will the data be manipulated  
to create the reports?  
(*required processing*)

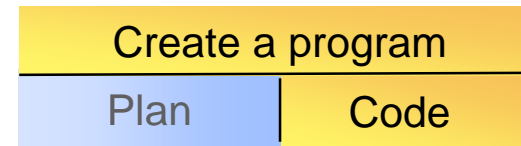


# The Programming Process

**Step 2:** Create a program (code).

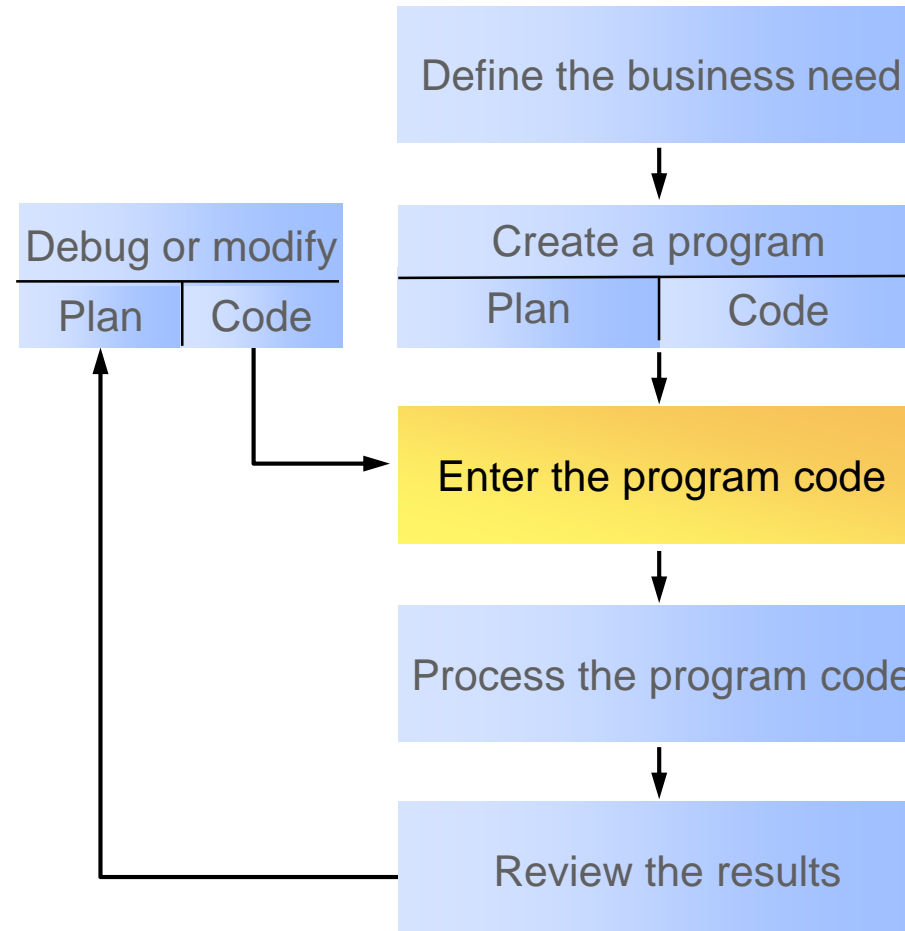
In the coding phase, you do the following:

- determine the computer language needed (*software*)
- use the program plan to write a set of instructions for the computer to process (*program code*)



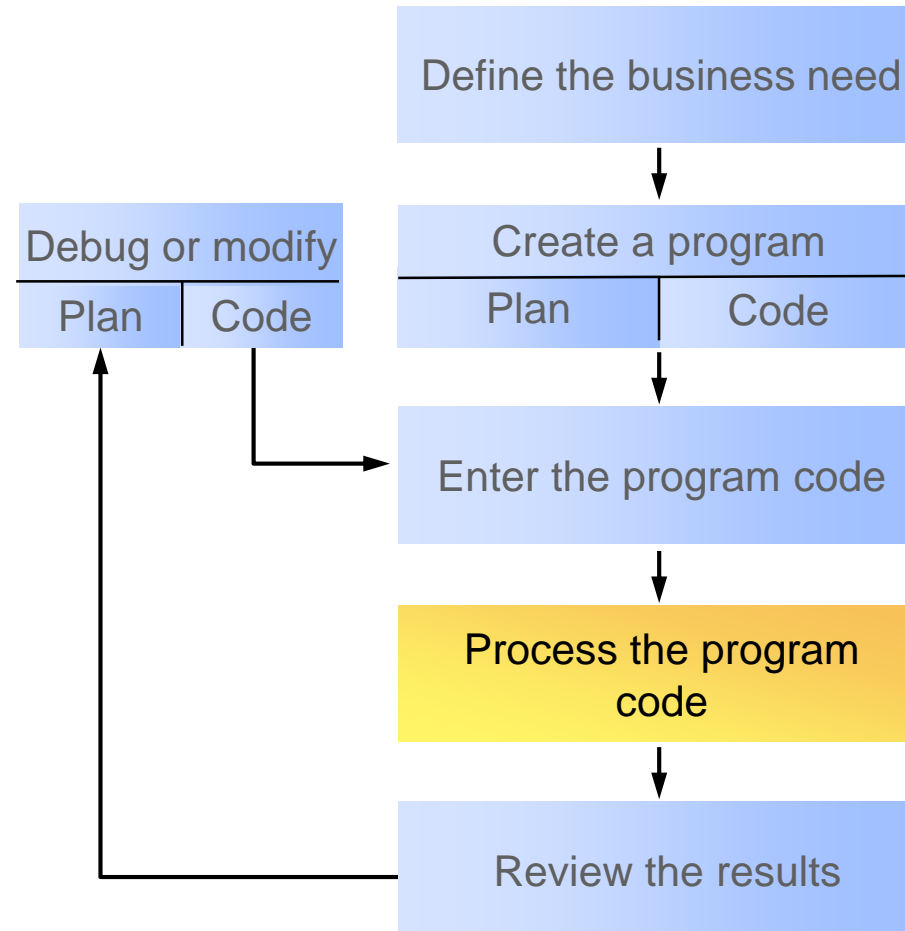
# The Programming Process

**Step 3:** Enter the program code.



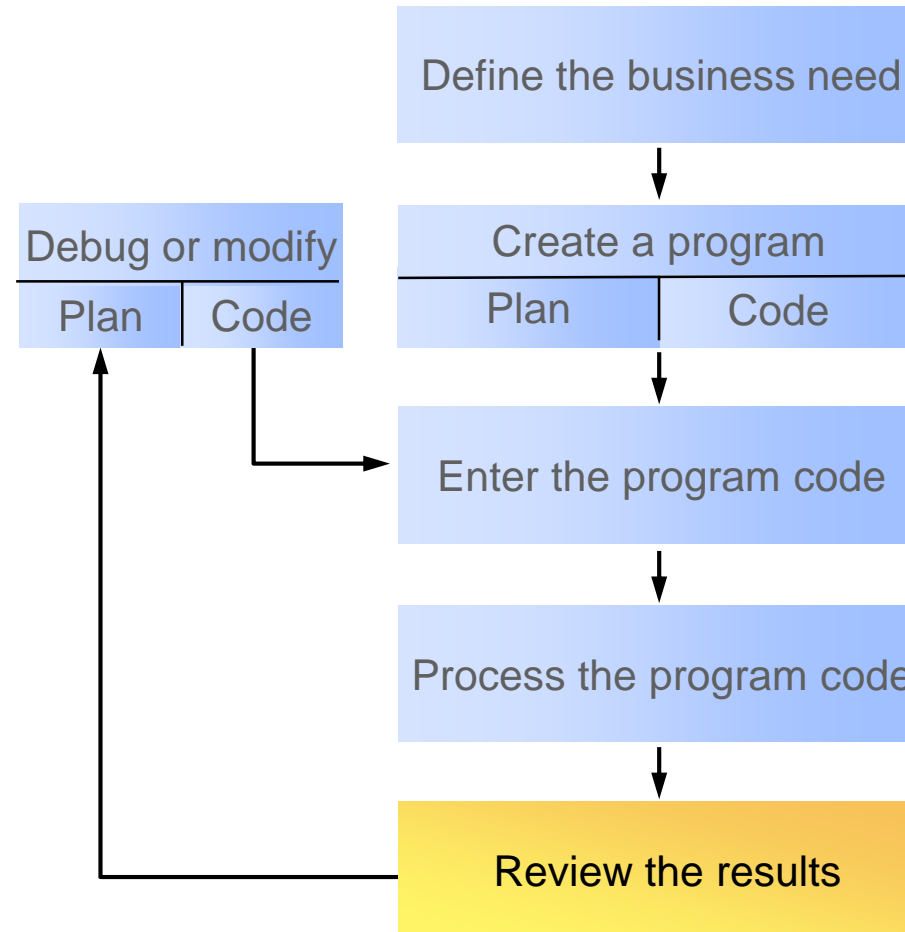
# The Programming Process

**Step 4:** Instruct the computer to process the code.



# The Programming Process

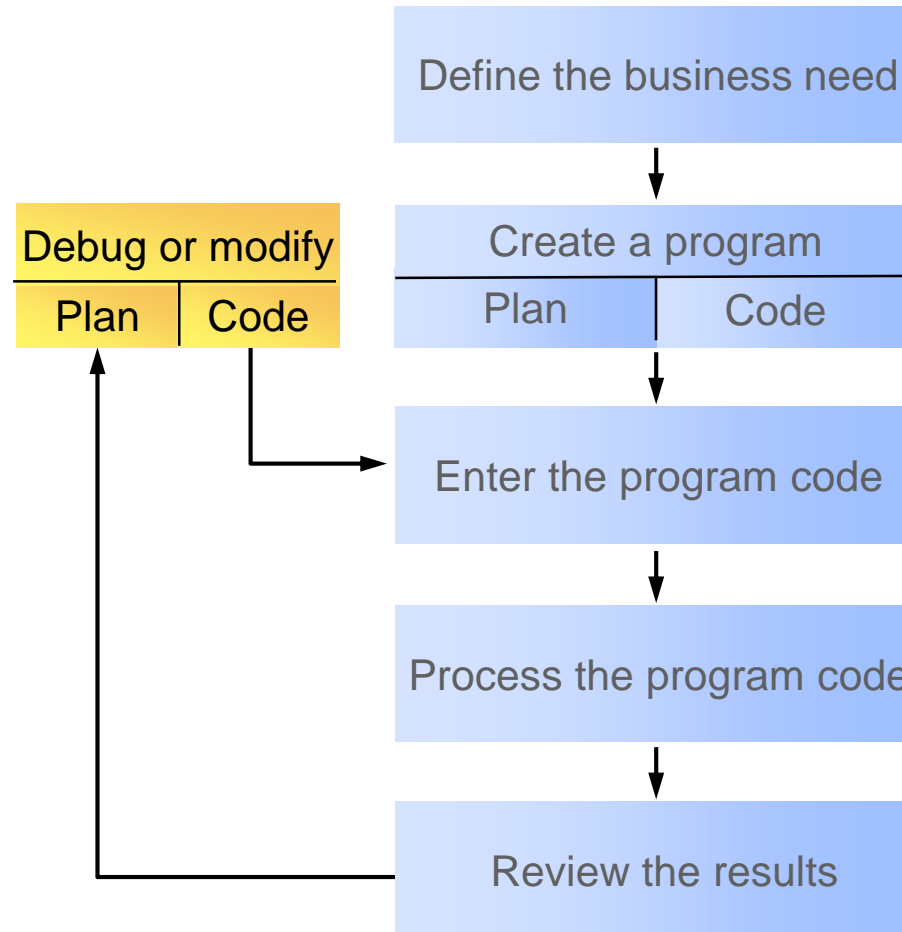
**Step 5:** Review the reports, messages, or output files.





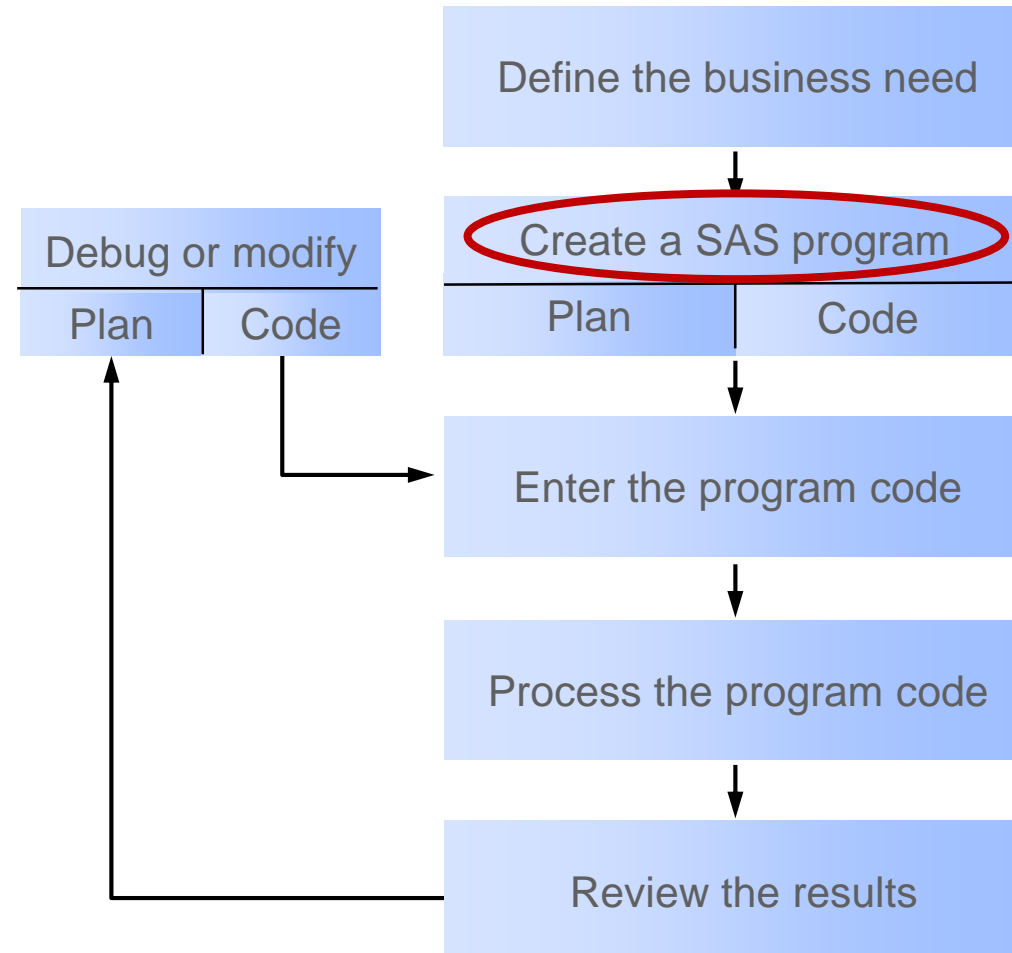
# The Programming Process

**Step 6:** Modify the code and repeat the process.

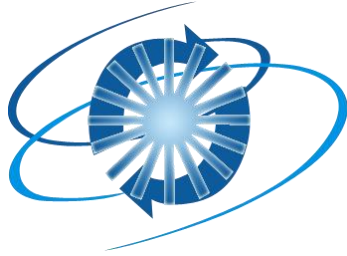


# The SAS Programming Process

The SAS programming process parallels the general programming process but uses SAS as the programming language.







## Exercise

---

This exercise reinforces the concepts discussed previously.

# Defining the Programming Process

1.a. Number the tasks in the order in which they should be performed.

_____	Enter the program code.
_____	Review the results.
_____	Process the program code.
_____	Debug or modify.
_____	Define the business need.
_____	Create a program.

# Defining the Programming Process

1.a. Number the tasks in the order in which they should be performed.

_____	Enter the program code.
_____	Review the results.
_____	Process the program code.
_____	Debug or modify.
<u>1</u>	Define the business need.
_____	Create a program.

# Defining the Programming Process

1.a. Number the tasks in the order in which they should be performed.

_____	Enter the program code.
_____	Review the results.
_____	Process the program code.
_____	Debug or modify.
<u>1</u>	Define the business need.
<u>2</u>	Create a program.

# Defining the Programming Process

1.a. Number the tasks in the order in which they should be performed.

<u>3</u>	Enter the program code.
<u>  </u>	Review the results.
<u>  </u>	Process the program code.
<u>  </u>	Debug or modify.
<u>1</u>	Define the business need.
<u>2</u>	Create a program.



# Defining the Programming Process

1.a. Number the tasks in the order in which they should be performed.

<u>3</u>	Enter the program code.
<u>  </u>	Review the results.
<u>4</u>	Process the program code.
<u>  </u>	Debug or modify.
<u>1</u>	Define the business need.
<u>2</u>	Create a program.

# Defining the Programming Process

1.a. Number the tasks in the order in which they should be performed.

<u>3</u>	Enter the program code.
<u>5</u>	Review the results.
<u>4</u>	Process the program code.
<u> </u>	Debug or modify.
<u>1</u>	Define the business need.
<u>2</u>	Create a program.

# Defining the Programming Process

1.a. Number the tasks in the order in which they should be performed.

<u>3</u>	Enter the program code.
<u>5</u>	Review the results.
<u>4</u>	Process the program code.
<u>6</u>	Debug or modify.
<u>1</u>	Define the business need.
<u>2</u>	Create a program.

# Defining the Programming Process

**1.b.** What are the two phases of creating a program?  
Circle the appropriate answer or answers.

- 1) testing
- 2) planning
- 3) coding
- 4) analyzing

# Defining the Programming Process

**1.b.** What are the two phases of creating a program?  
Circle the appropriate answer or answers.

- 1) testing
- ☒ 2) planning
- ☒ 3) coding
- 4) analyzing

# Chapter Review



1. Computer programs can be written to enter, read, and process data.

- ☐ True
- ☐ False

1. Computer programs can be written to enter, read, and process data.

- ☒ True
- ☐ False

**Computer programs are a set of instructions written by a user in a particular computer language in order to perform one or more tasks.**



2. Which SAS product are we working with in this class?
- a. Base SAS
  - b. SAS Basics
  - c. SAS Explorer
  - d. SAS Foundation

2. Which SAS product are we working with in this class?

- a. Base SAS
- b. SAS Basics
- c. SAS Explorer
- d. SAS Foundation

**We will be programming with Base SAS software.  
SAS Basics is not a product. SAS Foundation is a set  
of tools that includes Base SAS.**

3. The programming process is ***not*** iterative in nature.

- ☐ True
- ☐ False

3. The programming process is ***not*** iterative in nature.

☐ True

☒ False

**The programming process is definitely an iterative process. After you create and process a program, you might need to repeat the steps previously performed to correct errors, add to the program code, and so on.**

# Chapter 1: Introduction

**1.1 Overview**

**1.2 Introduction to programming**

**1.3 The Programming Process**

**1.4 Course Logistics**

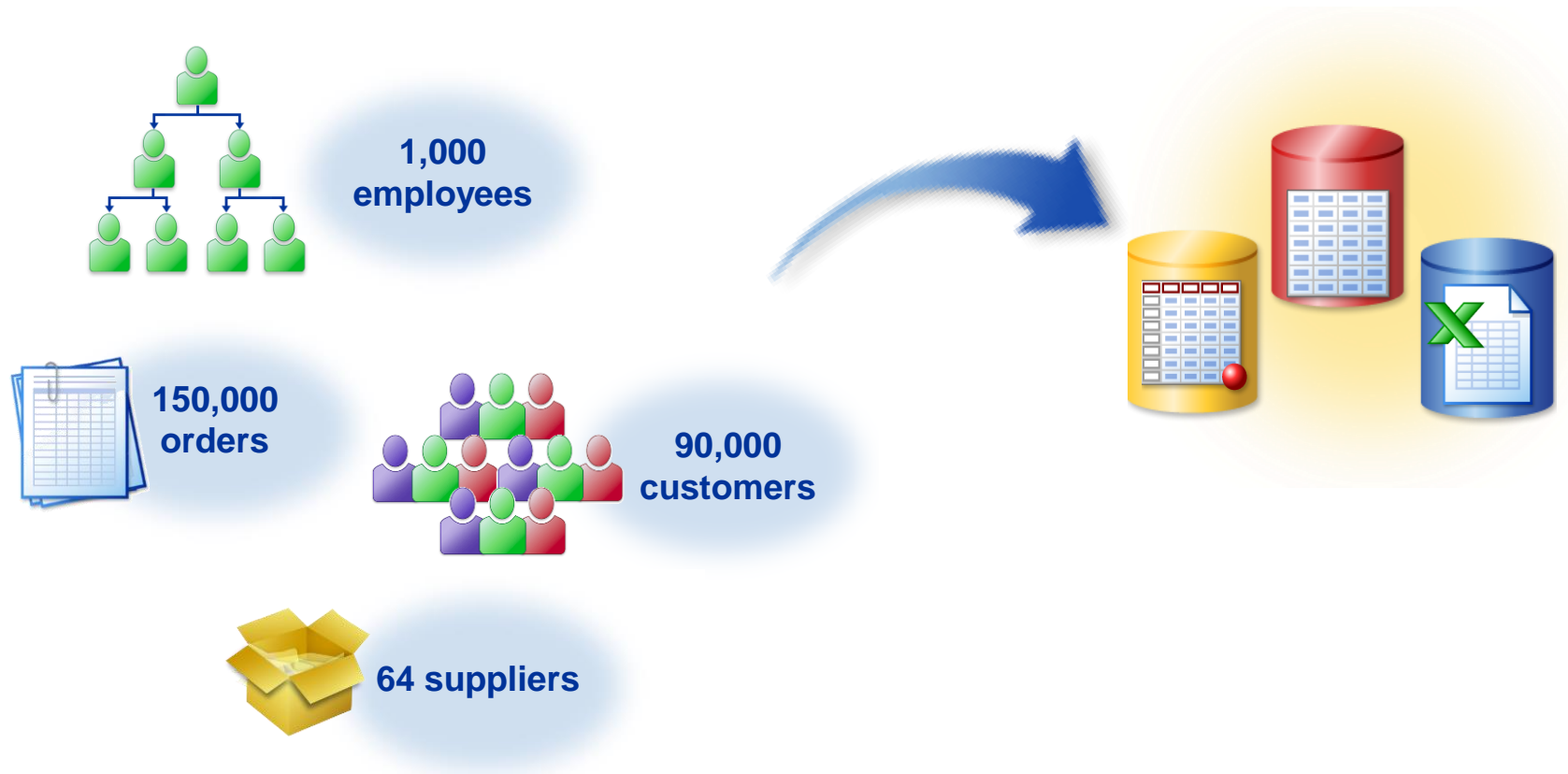
# Orion Star Sports & Outdoors

This course focuses 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.



# Program Naming Conventions

In this course, you use the structure below to retrieve and save SAS programs.

- ① course ID
- ② chapter #
- ③ type
  - a=activity
  - d=demo
  - e=exercise
  - s=solution
- ④ item #
- ⑤ placeholder

p104d01x

① ② ③ ④ ⑤



Programming 1, Chapter 4, Demo 1





# Chapter 2: SAS® Programs

**2.1 Introduction to SAS Programs**

**2.2 Submitting a SAS Program**

**2.3 SAS Program Syntax**

# Chapter 2: SAS® Programs

## 2.1 Introduction to SAS Programs

## 2.2 Submitting a SAS Program

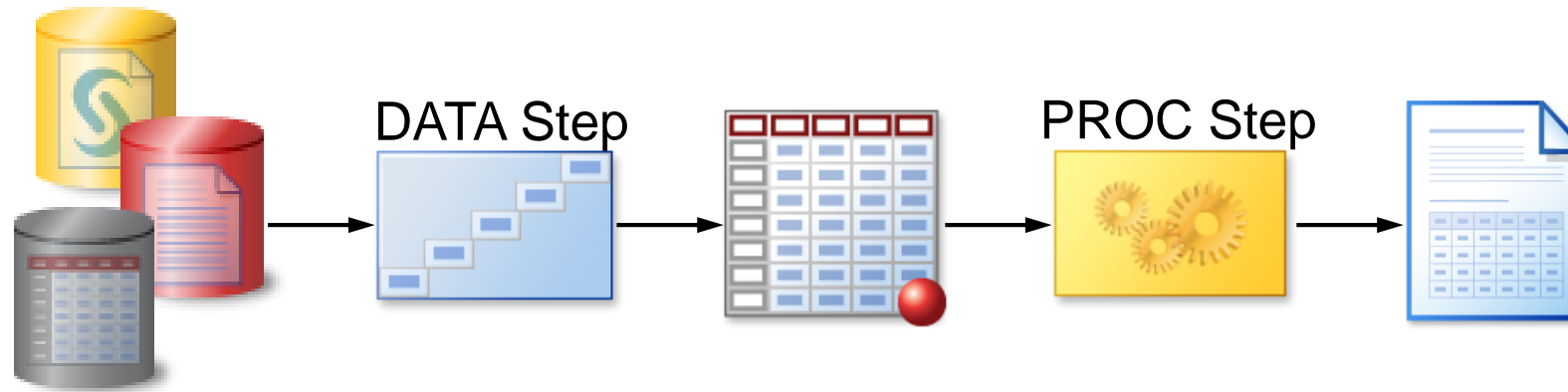
## 2.3 SAS Program Syntax

# Objectives

- List the components of a SAS program.

# SAS Programs

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



*There are only two types of steps in SAS:*

- *DATA steps* typically read from an input source and 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)

## 2.01 Short Answer Poll

How many steps are in program **p102d01**?

```
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;
```

## 2.01 Short Answer Poll – Correct Answer

How many steps are in program **p102d01**? **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;
```

DATA Step



PROC Step



PROC Step



p102d01



# 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;
```

p102d01

# 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;
```

# SAS Program Example

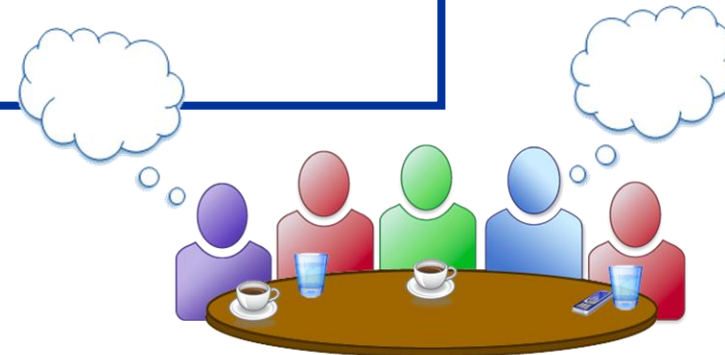
This PROC MEANS step summarizes the **Salary** variable in the **work.newsalesemps** data set.

```
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;
```

# Idea Exchange

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

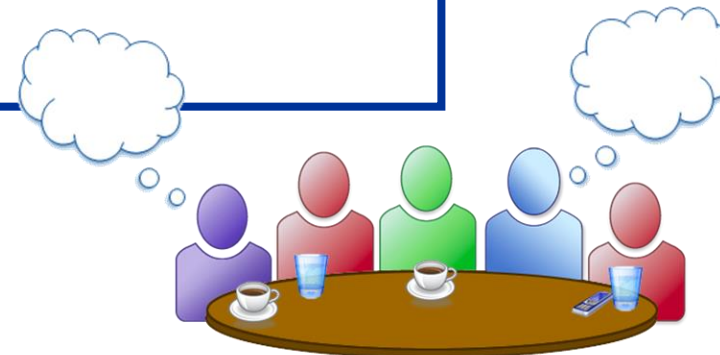


# Idea Exchange

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

the RUN statements





# Chapter 2: SAS® Programs

**2.1 Introduction to SAS Programs**

**2.2 Submitting a SAS Program**

**2.3 SAS Program Syntax**

# Processing Modes

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

<b><i>Interactive Mode</i></b>	<b>A SAS program is submitted within a SAS interface</b> for foreground processing
<b><i>Batch Mode</i></b>	A SAS program is submitted to the operating environment for background processing

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



# SAS Interfaces

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



SAS  
Windowing  
Environment



SAS  
Enterprise  
Guide



SAS  
Studio

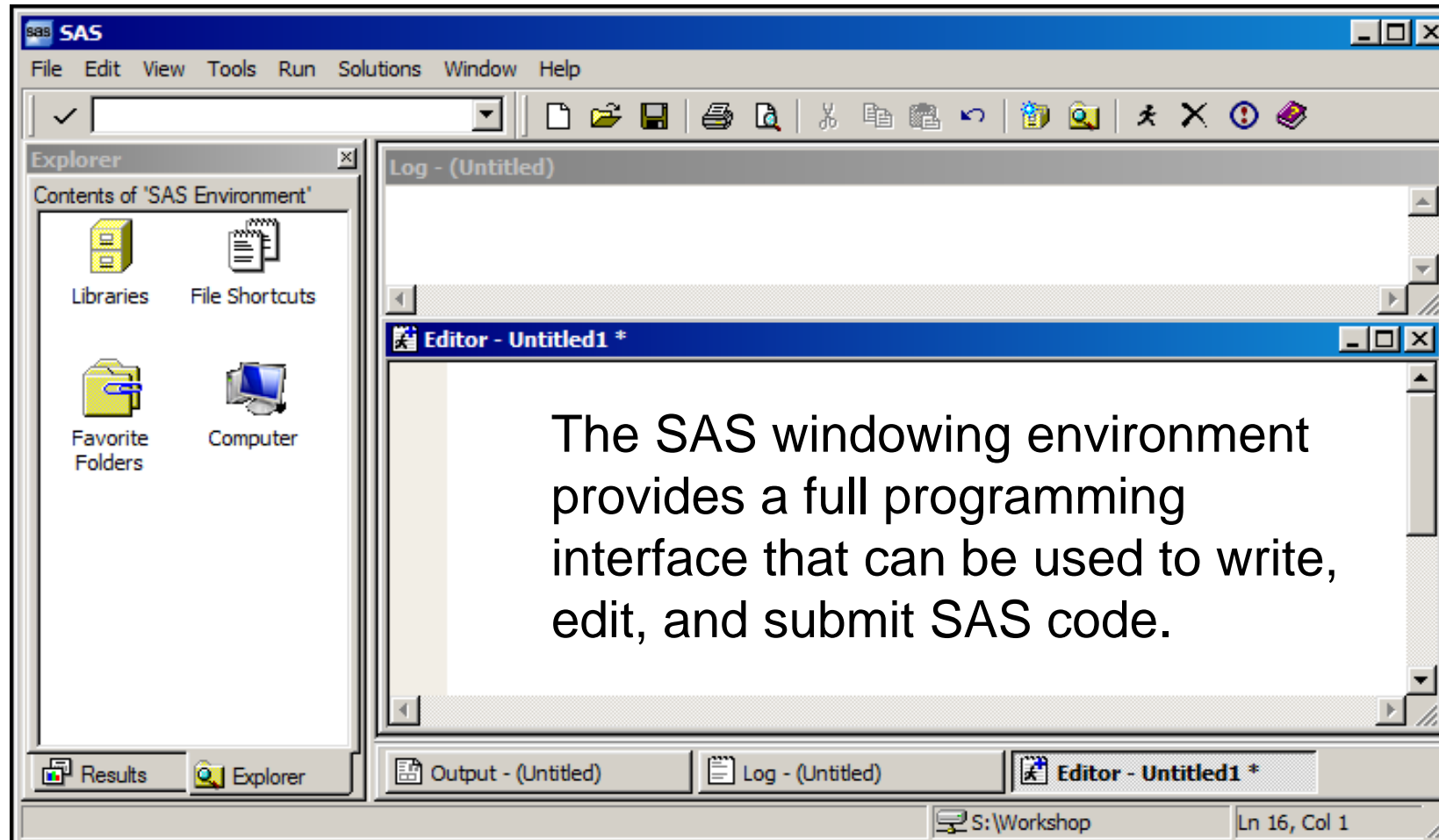
# SAS Interfaces

You will use SAS windowing environment interface during this course.





# SAS Windowing Environment



# SAS Interface Tabs or Windows

There are three primary tabs or windows.

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

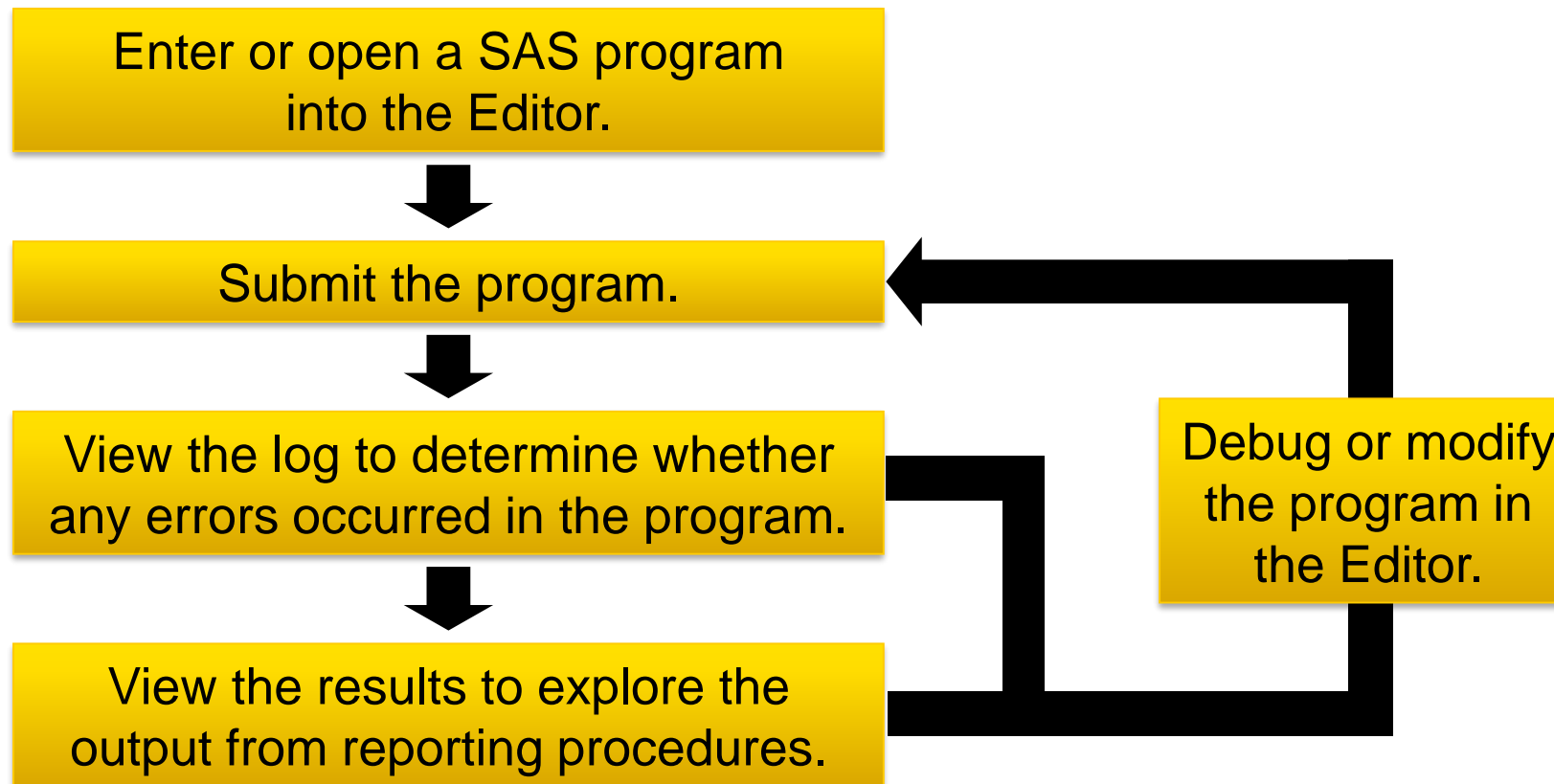
# SAS Interface Tabs or Windows

The SAS Windowing Environment interface uses specific terms to reference these three primary tabs or windows.

	<b>SAS Windowing Environment (windows)</b>
<b>Editor</b>	Enhanced or Program Editor
<b>Log</b>	Log
<b>Results</b>	Results Viewer or Output

# SAS Interface Tabs or Windows

The following is an example workflow of how a user might use the three primary tabs or windows:



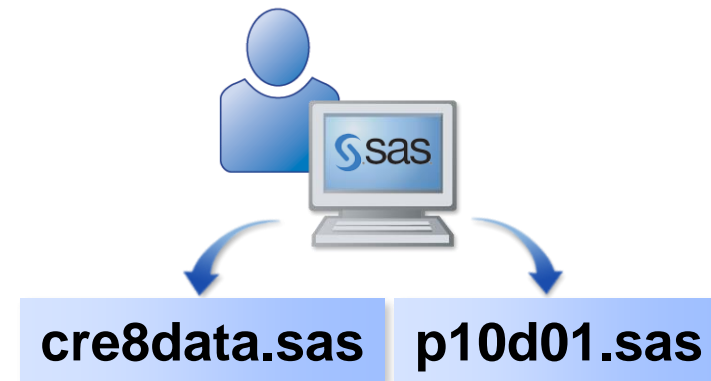


# Submitting SAS Programs in the SAS Windowing Environment

This demonstration illustrates how to use the Editor, Log, and Results Viewer windows within the SAS windowing environment.

In this section, the following actions are completed:

- Submit a SAS program to create the data files that are needed for the course.
- Submit the SAS program introduced in Section 2.1.





# Submitting SAS Programs in the SAS Windowing Environment

## Start the SAS windowing environment\*

By default, the SAS windowing environment consists of:

- a **left pane** containing the Results and Explorer windows
- a **right pane** containing the Output, Log, and Editor windows.


\*If you see a pop-up window that references a change notice or getting started with SAS, click Close.





# Submitting SAS Programs in the SAS Windowing Environment

## ■ Submitting the cre8data Program

Click  (**Open**) when the Editor window is the active window or select **File** ⇒ **Open Program**.


In the Open window, navigate in the file structure to find **cre8data** and click **Open**.

In the Editor window for the **cre8data** program, find the %LET statement.


If your data files are to be created at a location other than **s:\workshop**, change the value that is assigned to the **path** macro variable to reflect the location. If your data files are created in **s:\workshop**, then no change is needed.



The **cre8data** program uses forward slashes for portability across operating environments. UNIX and Linux require forward slashes. Windows accepts forward slashes and might convert them to backslashes.

Click  (**Submit**) or press F3 to submit the program.


In the Results Viewer window, verify that the output contains a list of data files.

Go to the Log window and click  (**New**) or select **Edit** ⇒ **Clear All** to clear the Log window.




# Submitting SAS Programs in the SAS Windowing Environment

## ■ Submitting the p102d01 Program

Click  (**Open**) when the Editor window is the active window or select **File** ⇒ **Open Program**.

In the Open window, navigate in the file structure to find **p102d01** and click **Open**.

Click  (**Submit**) or press F3 to submit the program.

In the Results Viewer window, view the PROC PRINT and PROC MEANS output.

Go to the Log window and verify that no errors or warnings appear. Be sure to scroll up to see all messages.

# Chapter 2: SAS® Programs

**2.1 Introduction to SAS Programs**

**2.2 Submitting a SAS Program**

**2.3 SAS Program Syntax**

# Objectives

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

# Business Scenario

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;
```

p102d01

## 2.03 Short Answer Poll

How many statements make up this DATA step?

```
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;
```

## 2.03 Short Answer Poll – Correct Answer

How many statements make up this DATA step?

```
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;
```

**This DATA step has five statements.**



# 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;
```

p102d02

This program is syntactically correct but difficult to read.

SAS is very flexible about spacing and program structure, but the flexibility can lead to programs that are difficult to read.

How many steps are in this program? It is the same program as the previous with 3 steps

# 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.

White space can be blanks, tabs, and new lines. Add them as needed to increase the readability of the code.

```
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.  |  
*-----*;  
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 */;  
run;  
/*  
proc print data=work.newsalesemps;  
run;  
*/  
proc means data=work.newsalesemps;  
    * var Salary ;  
run;
```

p102d03

## 2.04 Short Answer Poll

Open and examine **p102a01**. Based on the comments, which steps do you think are executed and what output is generated?

Submit the program. Which steps were executed?

## 2.04 Short Answer Poll – Correct Answer

Open and examine **p102a01**. Based on the comments, which steps do you think are executed and what output is generated?

Submit the program. Which steps were executed?

- **The DATA step executes and creates an output data set.**
- **The PROC PRINT step executes and produces a report.**
- **The PROC MEANS step is “commented out,” and therefore, does not execute.**





# Business Scenario

SAS programmers must be able to identify and correct syntax errors in a SAS program.



# 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

## 2.05 Short Answer Poll

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;
```

p102d04

invalid option

## 2.05 Short Answer Poll – Correct Answer

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

```
daat work newemps;  
length $ 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;
```

**misspelled keyword**


**missing semicolon**

**invalid option**

p102d04

# 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;
```

p102d04

Notice that the misspelled word D-A-A-T is displayed in red. This misspelling affects other statements following it since those statements are only permitted in a DATA step, and this is not recognized as such.

# 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.**



# Diagnosing and Correcting Syntax Errors

---

This demonstration illustrates how to diagnose and correct syntax errors.

## 2.06 Short Answer Poll

What is the syntax error 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  
run;  
  
proc means data=work.newsalesemps average min;  
  var Salary;  
run;
```

p102d05



## 2.06 Short Answer Poll – Correct Answer

What is the syntax error 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  
run;  
  
proc means data=work.newsalesemps average min;  
  var Salary;  
run;
```

p102d05

**The program contains unbalanced quotation marks in the DLM= option in the INFILE statement.**

## 2.06 Correcting quotation mark


### 14. Correcting Quotation Marks in the SAS Windowing Environment

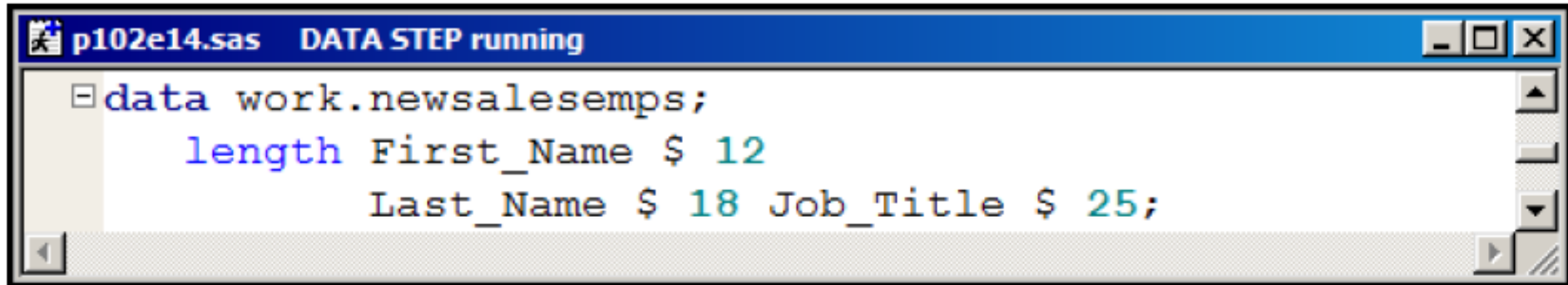
- a. In the SAS windowing environment, click  (**Open**) when the Editor window is the active window or select **File** ⇒ **Open Program**.
- b. In the Open window, navigate in the file structure to find **p102e14** and click **Open**. Notice that the closing quotation mark for the DLM= option in the INFILE statement is missing.
- c. Click  (**Submit**) or press F3 to submit the program.
- d. In the Log window, notice that there are no messages following each step in the log. The absence of messages often indicates unbalanced quotation marks.

#### Partial SAS Log

```
71      data work.newsalesemps;
72          length First_Name $ 12
73                  Last_Name $ 18 Job_Title $ 25;
74          infile "&path\\newemps.csv" dlm=',';
75          input First_Name $ Last_Name $
76                  Job_Title $ Salary;
77      run;
78
79      proc print data=work.newsalesemps;
80      run;
81
82      proc means data=work.newsalesemps;
83          var Salary;
84      run;
```

## 2.06 Correcting quotation mark

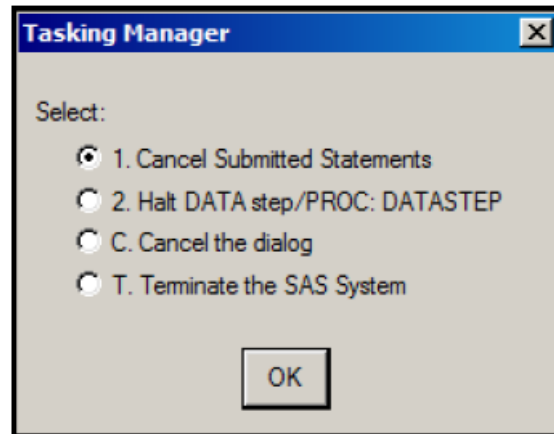
- e. In the Log window, click  (New) or select **Edit** ⇒ **Clear All** to clear the Log window.
- f. Return to the program. Notice the “DATA STEP running” message in the banner of the Editor window. This message appears because the RUN statement was viewed as part of the character literal and not as a step boundary.



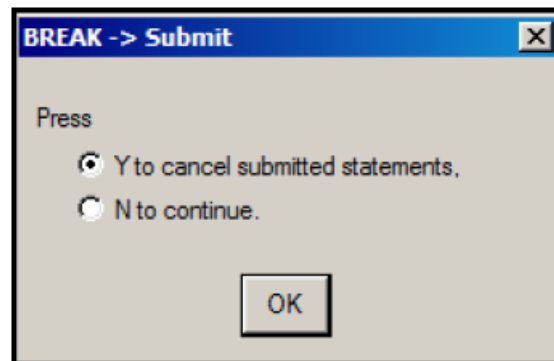
```
data work.newsalesemps;  
    length First_Name $ 12  
           Last_Name $ 18 Job_Title $ 25;
```

- g. To stop the DATA step from running, click  (Break) or press the Ctrl and Break keys.
- h. Select the **1. Cancel Submitted Statements** radio button in the Tasking Manager window. Click **OK**

## 2.06 Correcting quotation mark



- i. Select the **Y to Cancel submitted statements.** radio button. Click **OK**.



- j. Add a closing quotation mark to the DLM= option in the INFILE statement to correct the program.

```
infile "&path\newemps.csv" dlm=' , ' ;
```

# Chapter Review



2. Which of the following is a SAS syntax requirement?
- a. Begin each statement in column one.
  - b. Put only one statement on each line.
  - c. Separate each step with a line space.
  - d. End each statement with a semicolon.
  - e. Put a RUN statement after every DATA or PROC step.

2. Which of the following is a SAS syntax requirement?
- a. Begin each statement in column one.
  - b. Put only one statement on each line.
  - c. Separate each step with a line space.
  - ☒ d. End each statement with a semicolon.
  - e. Put a RUN statement after every DATA or PROC step.

3. Which of the following steps is typically used to generate reports and graphs?
- a. DATA
  - b. PROC
  - c. REPORT
  - d. RUN



3. Which of the following steps is typically used to generate reports and graphs?
- a. DATA
  - ☒ b. PROC
  - c. REPORT
  - d. RUN

4. Does this comment contain syntax errors?

```
/*  
Report created for budget  
presentation; revised October 15.  
*/  
proc print data=work.newloan;  
run;
```

- a. No. The comment is correctly specified.
- b. Yes. Every comment line must end with a semicolon.
- c. Yes. The comment text incorrectly begins on line one.
- d. Yes. The comment contains a semicolon, which causes an error message.

4. Does this comment contain syntax errors?

```
/*  
Report created for budget  
presentation; revised October 15.  
*/  
proc print data=work.newloan;  
run;
```

- a. No. The comment is correctly specified.
- b. Yes. Every comment line must end with a semicolon.
- c. Yes. The comment text incorrectly begins on line one.
- d. Yes. The comment contains a semicolon, which causes an error message.

5. What result would you expect from submitting this step?

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

- a. an HTML report of the **work.newsalesemps** data set
- b. an error message in the log
- c. a LISTING report of the **work.newsalesemps** data set
- d. the creation of the temporary data set **work.newsalesemps**

5. What result would you expect from submitting this step?

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

- a. an HTML report of the **work.newsalesemps** data set
- ☒ b. an error message in the log
- c. a LISTING report of the **work.newsalesemps** data set
- d. the creation of the temporary data set **work.newsalesemps**

6. If you submit a program containing unbalanced quotation marks in SAS windowing environment, you can simply correct the error and resubmit the program.
- a. True
  - b. False

6. If you submit a program containing unbalanced quotation marks in SAS windowing environment, you can simply correct the error and resubmit the program.

a. True

☒ b. False

SAS keeps a running total of the number of quotation marks in your code. In the windowing environment, an odd number of quotation marks would cause the program to hang; you would have to stop the program before adding the missing quotation mark and resubmitting the program

7. What happens if you submit the following program?

```
porc print data=work.newsalesemps;  
run;
```

- a. SAS does not execute the step.
- b. SAS assumes that the keyword PROC is misspelled and executes the PROC PRINT step.



7. What happens if you submit the following program?

```
porc print data=work.newsalesemps;  
run;
```

- a. SAS does not execute the step.
- ☒ b. SAS assumes that the keyword PROC is misspelled and executes the PROC PRINT step.

8. Suppose you submit a short, simple DATA step. If the active window displays the message **DATA step running** for a long time, what probably happened?
- a. You misspelled a keyword.
  - b. You forgot to end the DATA step with a RUN statement.
  - c. You specified an invalid data set option.
  - d. Some data values were not appropriate for the SAS statements that you specified.

8. Suppose you submit a short, simple DATA step. If the active window displays the message **DATA step running** for a long time, what probably happened?
- a. You misspelled a keyword.
  - ☒ b. You forgot to end the DATA step with a RUN statement.
  - c. You specified an invalid data set option.
  - d. Some data values were not appropriate for the SAS statements that you specified.

# Chapter 3: Accessing Data

## 3.1 Examining SAS Data Sets

## 3.2 Accessing SAS Libraries

# Chapter 3: Accessing Data

## 3.1 Examining SAS Data Sets

## 3.2 Accessing SAS Libraries

# Objectives

- Define the components of a SAS data set.
- Use the CONTENTS procedure to browse the descriptor portion of a SAS data set.
- Use the PRINT procedure to browse the data portion of a SAS data set.
- Define a SAS variable.
- Define a missing value.
- Define a SAS date value.

# Business Scenario

Many SAS data sets related to the Orion Star project already exist. The programmers need to know how to display the structure and contents of the data sets.

SAS Data Set

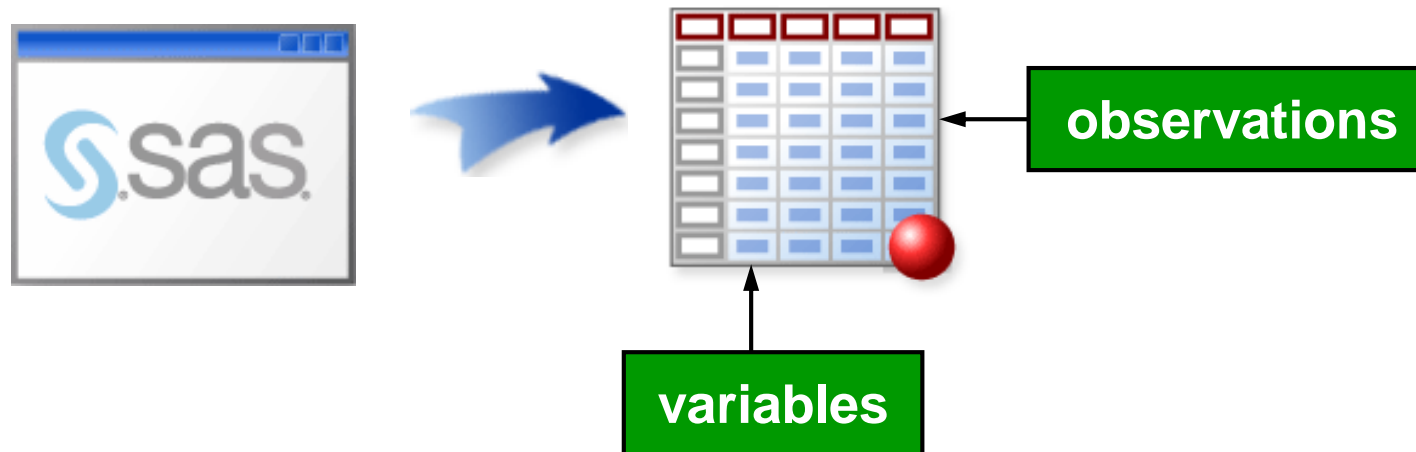


Report



# What Is a SAS Data Set?

A *SAS data set* is a specially structured data file that SAS creates and that only SAS can read. A SAS data set is a table that contains observations and variables.



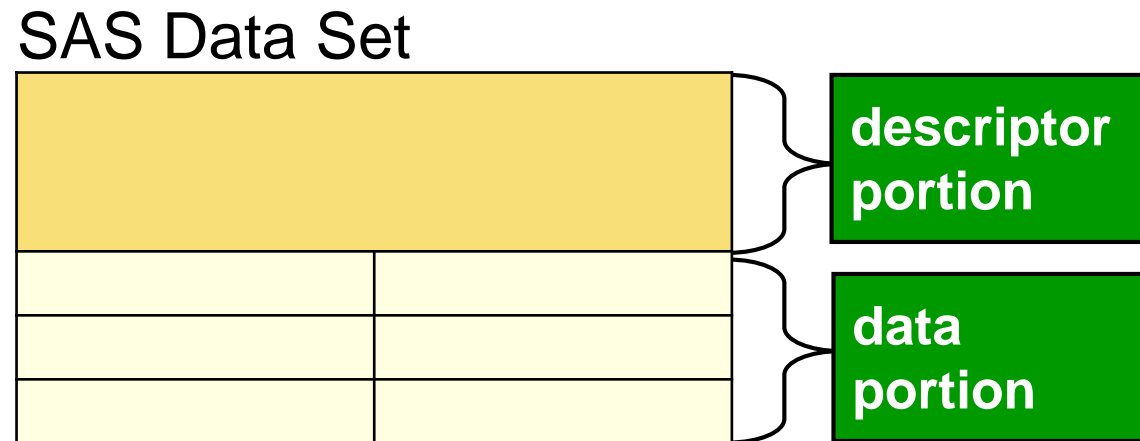


# SAS Data Set Terminology

SAS Terminology		Database Terminology	
SAS Data Set	↔	Table	
Observation	↔	Row	
Variable	↔	Column	

# SAS Data Set Terminology

A SAS data set contains a descriptor portion and a data portion.





# Descriptor Portion

The *descriptor portion* contains the following metadata:

- general properties (such as data set name and number of observations)
- variable properties (such as name, type, and length)

## Partial **work.newsalesemps**

<b>Data Set Name</b>		<b>WORK.NEWSALESEMPs</b>			<b>general properties</b>
<b>Engine</b>		<b>V9</b>			
<b>Created</b>		<b>Mon, Feb 27, 2012 01:28 PM</b>			
<b>Observations</b>		<b>71</b>			
<b>Variables</b>		<b>4</b>			
...					<b>variable properties</b>
<b>First_Name</b>	<b>Last_Name</b>	<b>Job_Title</b>	<b>Salary</b>		
<b>\$ 12</b>	<b>\$ 18</b>	<b>\$ 25</b>	<b>N 8</b>		

# Browsing the Descriptor Portion

Use *PROC CONTENTS* to display the descriptor portion of a SAS data set.

p103d01

```
proc contents data=work.newsalesemps;  
run;
```

```
PROC CONTENTS DATA=SAS-data-set;  
RUN;
```

# Viewing the Output

## Partial PROC CONTENTS Output

The CONTENTS Procedure			
Data Set Name	WORK.NEWSALESEMPS	Observations	71
Member Type	DATA	Variables	4
Engine	V9	Indexes	0
Created	Mon, Feb 27, 2012 01:28:51 PM	Observation Length	64
Last Modified	Mon, Feb 27, 2012 01:28:51 PM	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Engine/Host Dependent Information			
...			
Alphabetic List of Variables and Attributes			
#	Variable	Type	Len
1	First_Name	Char	12
3	Job_Title	Char	25
2	Last_Name	Char	18
4	Salary	Num	8

## 3.01 Short Answer Poll

Open program **p103a01**. Add a PROC CONTENTS step after the DATA step to view **work.donations**. Submit the program and review the results. How many observations are in the data set **work.donations**? **124 observations**

## 3.01 Short Answer Poll – Correct Answer

Open program **p103a01**. Add a PROC CONTENTS step after the DATA step to view **work.donations**. Submit the program and review the results. How many observations are in the data set **work.donations**? **124 observations**

```
data work.donations;  
    infile "&path\donation.dat";  
    input Employee_ID Qtr1 Qtr2 Qtr3 Qtr4;  
    Total=sum(Qtr1,Qtr2,Qtr3,Qtr4);  
run;  
  
proc contents data=work.donations;  
run;
```

# Data Portion

The *data portion* of a SAS data set contains the data values, which are either character or numeric.

Partial **work.newsalesemps**

First_Name	Last_Name	Job_Title	Salary	variable names
Satyakam	Denny	Sales Rep. II	26780	
Monica	Kletschkus	Sales Rep. IV	30890	data values
Kevin	Lyon	Sales Rep. I	26955	
Petrea	Soltau	Sales Rep. II	27440	

character values

numeric values



# Browsing the Data Portion

Use *PROC PRINT* to display the data portion of a SAS data set.

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

```
PROC PRINT DATA=SAS-data-set;  
RUN;
```

# Viewing the Output

## Partial PROC PRINT Output

Obs	First_Name	Last_Name	Job_Title	Salary
1	Satyakam	Denny	Sales Rep. II	26780
2	Monica	Kletschkus	Sales Rep. IV	30890
3	Kevin	Lyon	Sales Rep. I	26955
4	Petrea	Soltau	Sales Rep. II	27440
5	Marina	Iyengar	Sales Rep. III	29715

# SAS Variable Names

SAS variable names

- can be 1 to 32 characters long.
- must start with a letter or underscore. Subsequent characters can be letters, underscores, or numerals. (no special characters)
- can be uppercase, lowercase, or mixed case.
- are not case sensitive.

Salary

\_score2\_

cust\_ID

month1

FirstName

## 3.02 Multiple Answer Poll

Which variable names are invalid?

- a. data5mon
- b. 5monthsdata
- c. data#5
- d. five months data
- e. five\_months\_data
- f. FiveMonthsData
- g. fivemonthsdata

## 3.02 Multiple Answer Poll – Correct Answer

Which variable names are invalid?

- a. data5mon
- ☒ b. 5monthsdata
- ☒ c. data#5
- ☒ d. five months data
- e. five\_months\_data
- f. FiveMonthsData
- g. fivemonthsdata

# Data Types

A SAS data set supports two types of variables.

## *Character variables*

- can contain any value: letters, numerals, special characters, and blanks
- range from 1 to 32,767 characters in length
- have 1 byte per character.

## *Numeric variables*

- store numeric values using floating point or binary representation
- have 8 bytes of storage by default
- can store 16 or 17 significant digits.

# Missing Data Values

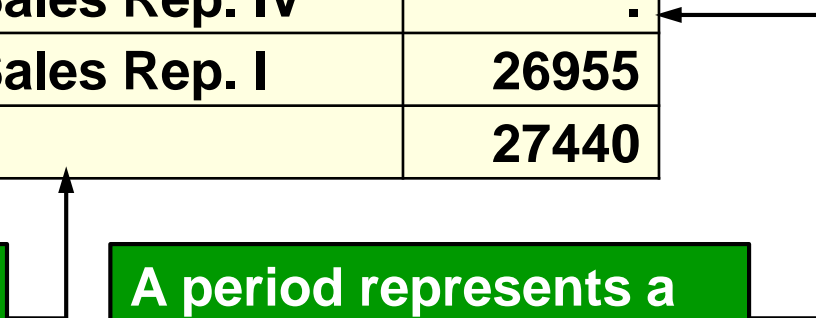
Missing values are valid values in a SAS data set.

Partial **work.newsalesemps**

First_Name	Last_Name	Job_Title	Salary
Monica	Kletschkus	Sales Rep. IV	.
Kevin	Lyon	Sales Rep. I	26955
Petrea	Soltau		27440

A blank represents a missing character value.

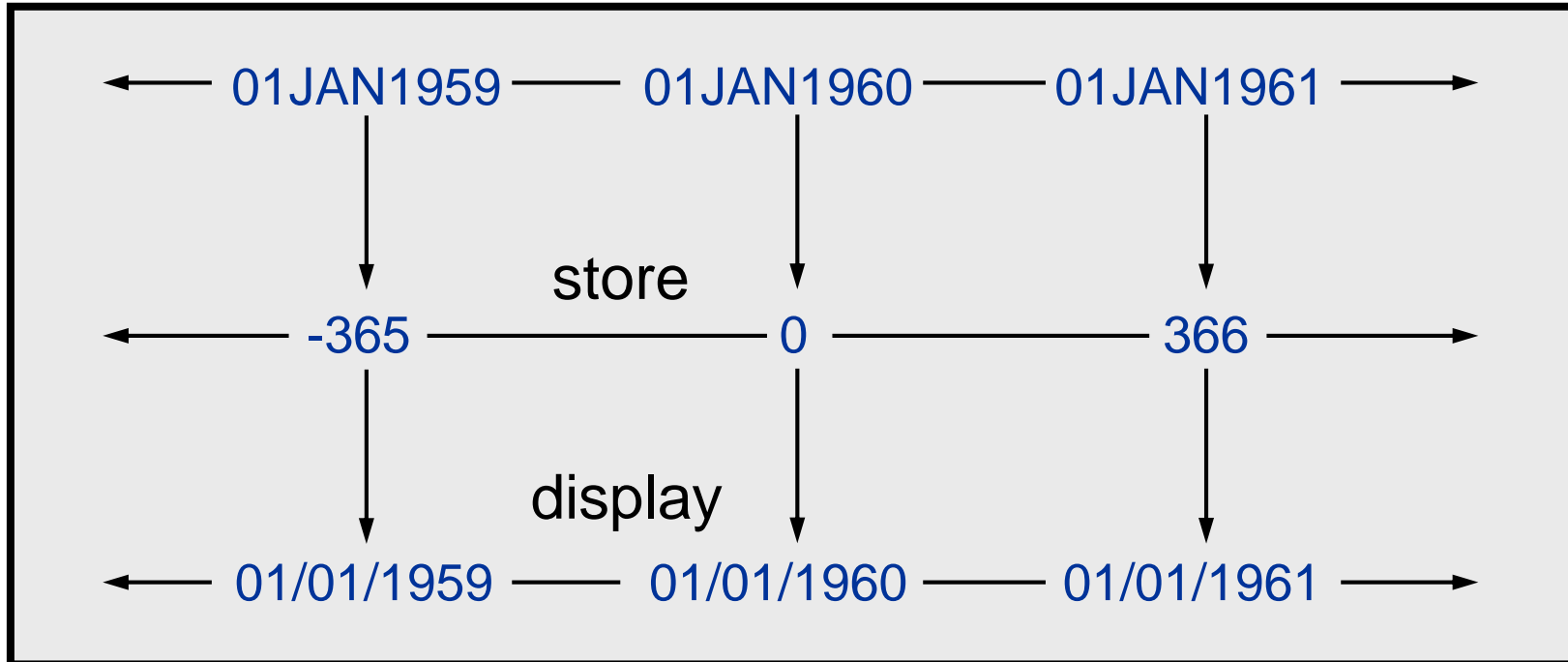
A period represents a missing numeric value.



 A value must exist for every variable in every observation.

# SAS Date Values

SAS stores calendar dates as numeric values.



A *SAS date value* is stored as the number of days between January 1, 1960, and a specific date.



## 3.03 Short Answer Poll

Submit program **p103a02**. View the output to retrieve the current date as a SAS date value (that is, a numeric value referencing January 1, 1960). What is the numeric value for today's date?

## 3.03 Short Answer Poll – Correct Answer

Submit program **p103a02**. View the output to retrieve the current date as a SAS date value (that is, a numeric value referencing January 1, 1960). What is the numeric value for today's date?

**The answer depends on the current date.**

**Example: If the current date is November 9th, 2023,  
the numeric value is 23323**

```
data work.date;  
    CurrentDate=today();  
run;  
  
proc print data=work.date;  
run;
```



# Chapter 3: Accessing Data

## 3.1 Examining SAS Data Sets

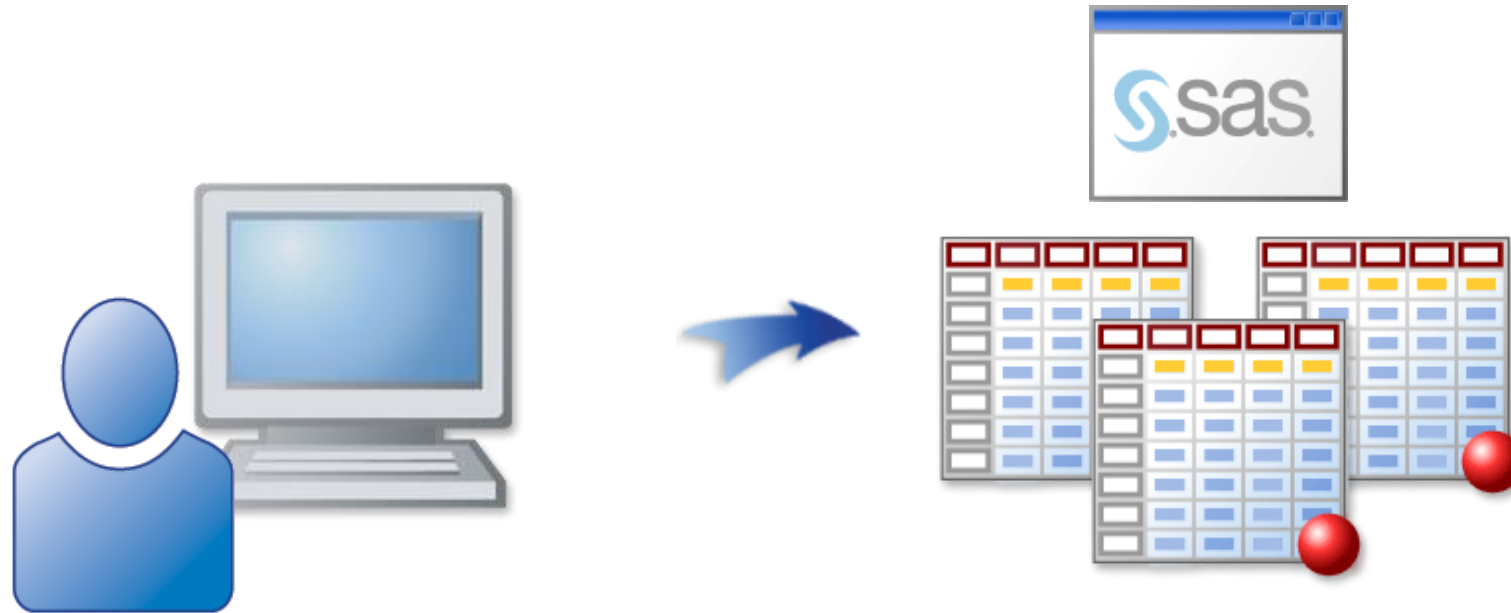
## 3.2 Accessing SAS Libraries

# Objectives

- Explain the concept of a SAS library.
- State the difference between a temporary library and a permanent library.
- Use a LIBNAME statement to assign a library reference name to a SAS library.
- Investigate a SAS library programmatically and interactively.

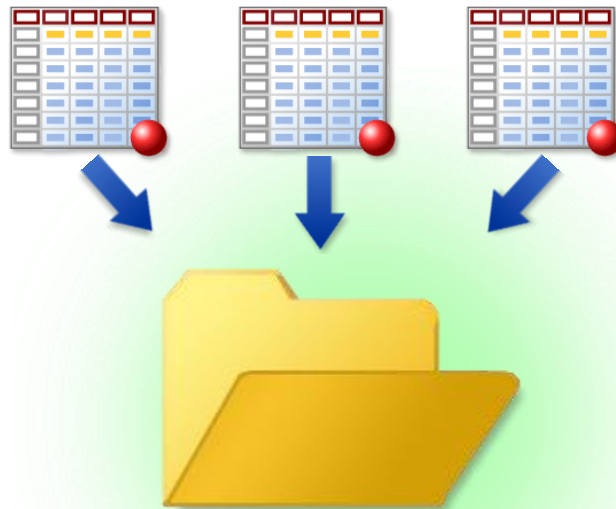
# Business Scenario

SAS programmers need to access existing SAS data sets, so they need to understand how the data sets are stored in SAS.



# SAS Libraries

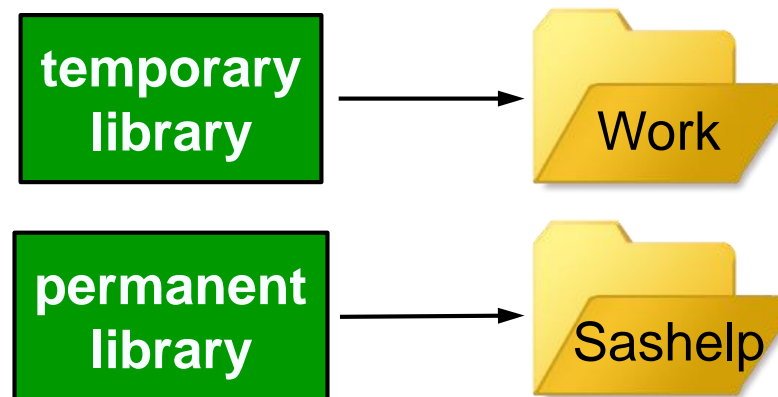
SAS data sets are stored in *SAS libraries*. A SAS library is a collection of SAS files that are referenced and stored as a unit.



A file can be stored in a temporary or permanent library.

# How SAS Libraries Are Defined

When a SAS session starts, SAS creates one temporary and at least one permanent SAS library. These libraries are open and ready to be used.



You refer to a SAS library by a logical name called a library reference name, or *libref*.



# Temporary Library

**Work** is a temporary library where you can store and access SAS data sets for the duration of the SAS session. It is the default library.



SAS deletes the **Work** library and its contents when the session terminates.

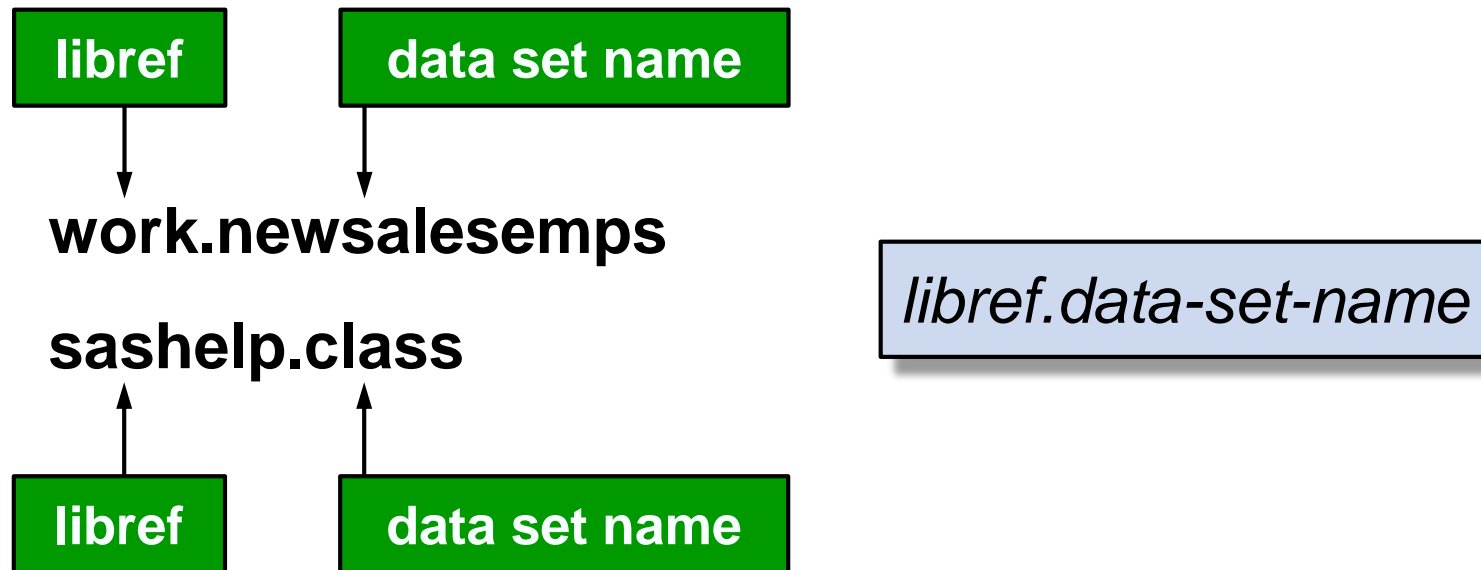
# Permanent Libraries

**Sashelp** is a permanent library that contains sample SAS data sets you can access during your SAS session.



# Accessing SAS Data Sets

All SAS data sets have a two-level name that consists of the libref and the data set name, separated by a period.



When a data set is in the temporary **Work** library, you can use a one-level name (for example, **newsalesemps**).

# Business Scenario

Orion Star programmers need to access and view SAS data sets that are stored in a permanent user-defined library.



You can access the system libraries or create your own. The libraries you create are permanent. If you can make a Windows folder, then you can create a SAS library.

# User-Defined Libraries

A user-defined library

- is created by the user.
- is permanent. Data sets are stored until the user deletes them.
- is not automatically available in a SAS session.
- is implemented within the operating environment's file system.

# User-Defined Libraries

Operating Environment	A SAS library is...	Example
Microsoft Windows	a folder	s:\workshop
UNIX	a directory	~/workshop

The user must submit a SAS LIBNAME statement to associate a libref with the physical location of the library.

# LIBNAME Statement

The SAS LIBNAME statement is a SAS statement.

```
libname orion "s:\workshop";
```

```
LIBNAME libref "SAS-library" <options>;
```

- It is not required to be in a DATA step or PROC step.
- It does not require a RUN statement.
- It executes immediately.
- It remains in effect until changed or canceled, or until the session ends.



Use the location of ***your*** course data in your LIBNAME statement.

# LIBNAME Statement

## Partial SAS Log

```
47  libname orion "s:\workshop";  
NOTE: Libref ORION was successfully assigned as follows:  
      Engine:          V9  
      Physical Name: s:\workshop
```

SAS files in **s:\workshop** are referenced using the **orion** libref.

*orion.data-set-name*



# Changing or Canceling a Libref

A libref remains in effect until you change or cancel it, or until you end your SAS session.

To change a libref, submit a LIBNAME statement with the same libref but a different path.

```
libname orion "c:\myfiles";
```

To cancel a libref, submit a LIBNAME statement with the CLEAR option.

```
libname orion clear;
```

## 3.04 Multiple Choice Poll

Which of the following correctly assigns the libref **myfiles** to a SAS library in the **c:\mysasfiles** folder?

- a. libname orion myfiles "c:\mysasfiles";
- b. libname myfiles "c:\mysasfiles";
- c. libref orion myfiles "c:\mysasfiles";
- d. libref myfiles "c:\mysasfiles";

## 3.04 Multiple Choice Poll – Correct Answer

Which of the following correctly assigns the libref **myfiles** to a SAS library in the **c:\mysasfiles** folder?

- a. libname orion myfiles "c:\mysasfiles";
- ☒ b. libname myfiles "c:\mysasfiles";
- c. libref orion myfiles "c:\mysasfiles";
- d. libref myfiles "c:\mysasfiles";

# Browsing a Library

You can browse a library

- programmatically using the CONTENTS procedure
- interactively using the Explorer window.



# Browsing a Library Programmatically

Use PROC CONTENTS with the `_ALL_` keyword to generate a list of all SAS files in a library.

p103d03

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

```
PROC CONTENTS DATA=libref._ALL_ NODS;  
RUN;
```

- `_ALL_` requests all the files in the library.
- The NODS option suppresses the individual data set descriptor information.
- NODS can be used only with the keyword `_ALL_`.

# Viewing the Output

## Partial PROC CONTENTS Output

The CONTENTS Procedure				
Directory				
		Libref	ORION	
		Engine	V9	
		Physical Name	S:\workshop	
		Filename	S:\workshop	
#	Name	Member Type	File Size	Last Modified
1	CHARITIES	DATA	9216	23Aug12:15:58:39
2	CONSULTANTS	DATA	5120	23Aug12:15:58:39
3	COUNTRY	DATA	17408	130ct10:19:04:39
	COUNTRY	INDEX	17408	130ct10:19:04:39
4	CUSTOMER	DATA	33792	04Nov11:09:52:27
5	CUSTOMER_DIM	DATA	33792	04Nov11:09:52:27

# Chapter Review



1. In which portion of a SAS data set are the following found?
  - the name of the data set
  - the type of the variable **Salary**
  - the creation date of the data set
  - a. descriptor portion
  - b. data portion



1. In which portion of a SAS data set are the following found?
  - the name of the data set
  - the type of the variable **Salary**
  - the creation date of the data set
  - a. descriptor portion
  - b. data portion

3. Which LIBNAME statement has the correct syntax?

- a. libname reports 's:\workshop';
- b. libname orion s:\workshop;
- c. libname 3456a 's:\workshop';

3. Which LIBNAME statement has the correct syntax?

- a. libname reports 's:\workshop';
- b. libname orion s:\workshop;
- c. libname 3456a 's:\workshop';

4. Which PROC step successfully prints a list of all data sets in the **orion** library without printing descriptor portions for the individual data sets?
- a. `proc contents data=orion.nods _all_;`  
`run;`
  - b. `proc contents data=orion._all_ nods;`  
`run;`
  - c. `proc print data=orion._all_ noobs;`  
`run;`
  - d. `proc print data=orion._all_ nods;`  
`run;`

4. Which PROC step successfully prints a list of all data sets in the **orion** library without printing descriptor portions for the individual data sets?

a. `proc contents data=orion.nods _all_;`  
`run;`

b. `proc contents data=orion._all_ nods;`  
`run;`

c. `proc print data=orion._all_ noobs;`  
`run;`

d. `proc print data=orion._all_ nods;`  
`run;`

5. In this data set, what type of variable is **Employee\_ID**?

- a. character
- b. numeric
- c. temporary
- d. missing

Obs	Employee_ID	Last	Salary
1	.	Ralston	29250
2	120101	Lu	163040
3	120104	Billington	46230
4	120105	Povey	27110
5	120106	Hornsey	.

5. In this data set, what type of variable is **Employee\_ID**?

- a. character
- ☒ b. numeric
- c. temporary
- d. missing

Obs	Employee_ID	Last	Salary
1	.	Ralston	29250
2	120101	Lu	163040
3	120104	Billington	46230
4	120105	Povey	27110
5	120106	Hornsey	.

6. What type of data set is the input data set in this PROC PRINT step?

```
proc print data=order_fact;  
run;
```

- a. temporary
- b. permanent
- c. There is not enough information to determine the type.



6. What type of data set is the input data set in this PROC PRINT step?

```
proc print data=order_fact;  
run;
```

- a. temporary
- b. permanent
- c. There is not enough information to determine the type.

8. Which of the following is not true of SAS date values?
- a. They are numeric.
  - b. They can be positive or negative values.
  - c. They represent the number of days between the day being stored and a base date.
  - d. The base date is January 1, 1900.

8. Which of the following is not true of SAS date values?
- a. They are numeric.
  - b. They can be positive or negative values.
  - c. They represent the number of days between the day being stored and a base date.
  - ☒ d. The base date is January 1, 1900.

The base date is January 1, 1960.

9. Which statement about SAS libraries is true?
- a. You refer to a SAS library by a logical name called a libname.
  - b. A SAS library is a collection of one or more SAS files that are referenced and stored as a unit.
  - c. A single SAS library can contain files that are stored in different physical locations.
  - d. At the end of each session, SAS deletes the contents of all SAS libraries.

9. Which statement about SAS libraries is true?
- a. You refer to a SAS library by a logical name called a libname.
  - ☒ b. A SAS library is a collection of one or more SAS files that are referenced and stored as a unit.
  - c. A single SAS library can contain files that are stored in different physical locations.
  - d. At the end of each session, SAS deletes the contents of all SAS libraries.