

STAT604

Lesson SAS 05

Portions Copyright © 2009 SAS Institute Inc., Cary, NC, USA. All rights reserved. Reproduced with permission of SAS Institute Inc., Cary, NC, USA. SAS Institute Inc. makes no warranties with respect to these materials and disclaims all liability therefor.

Chapter 9: Manipulating Data



9.1 Creating Variables

9.2 Creating Variables Conditionally

9.3 Subsetting Observations

Business Scenario

A new SAS data set named **Work.bonus** needs to be created by reading the **orion.sales** data set.

Work.bonus must include a new variable named **Bonus** that is equal to

- 500 for United States employees
- 300 for Australian employees.

Work.bonus must include another new variable named **Freq** that is equal to

- **Once a Year** for United States employees
- **Twice a Year** for Australian employees.

IF-THEN/ELSE Statements

Only **one** executable statement is allowed in IF-THEN/ELSE statements.

```
IF expression THEN statement;  
ELSE IF expression THEN statement;  
ELSE statement;
```

For the given business scenario, two statements need to be executed per each true expression.

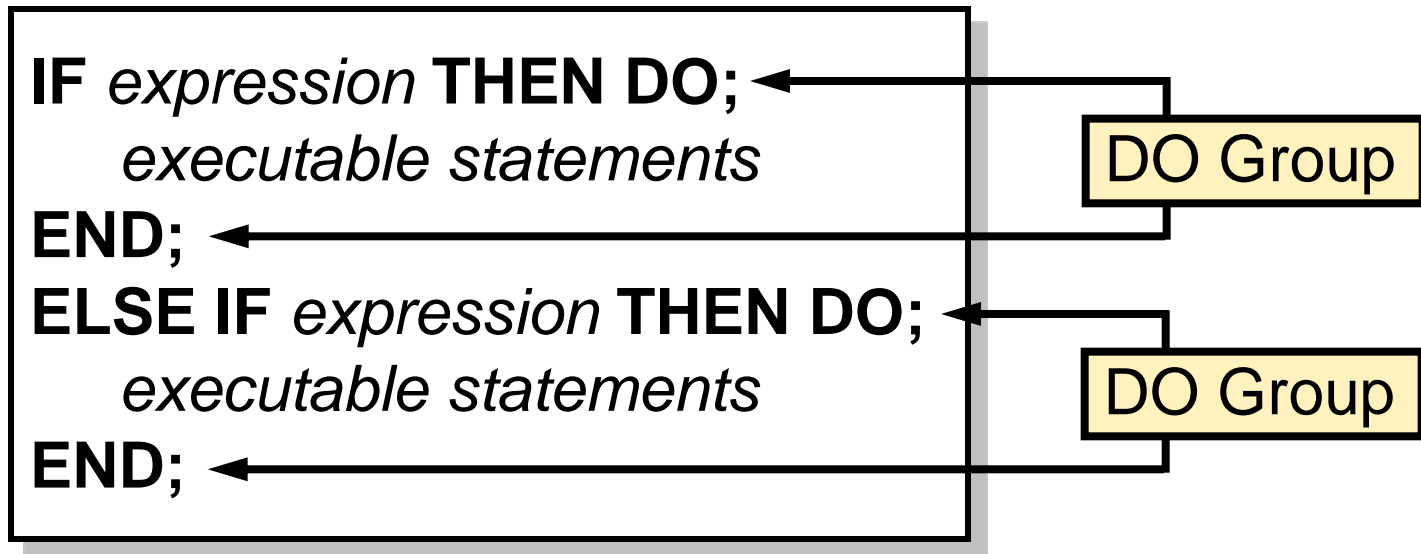
```
if Country='US' then
```

```
Bonus=500;
```

```
Freq='Once per Year';
```

IF-THEN DO/ELSE DO Statements

Multiple executable statements are allowed in IF-THEN DO/ELSE DO statements.



- Each DO group can contain multiple statements that apply to the expression.
- Each DO group ends with an END statement.

Business Scenario

Create another new variable named **Freq**.

```
data work.bonus;  
  set orion.sales;  
  if Country='US' then do;  
    Bonus=500;  
    Freq='Once a Year';  
  end;  
  else if Country='AU' then do;  
    Bonus=300;  
    Freq='Twice a Year';  
  end;  
run;
```

Business Scenario

```
proc print data=work.bonus;  
  var First Name Last Name  
      Country Bonus Freq;  
run;
```

Partial PROC PRINT Output

Obs	First_Name	Last_Name	Country	Bonus	Freq
60	Billy	Plested	AU	300	Twice a Yea
61	Matsuoka	Wills	AU	300	Twice a Yea
62	Vino	George	AU	300	Twice a Yea
63	Meera	Body	AU	300	Twice a Yea
64	Harry	Highpoint	US	500	Once a Year
65	Julienne	Magolan	US	500	Once a Year
66	Scott	Desanctis	US	500	Once a Year
67	Cherda	Ridley	US	500	Once a Year
68	Priscilla	Farren	US	500	Once a Year
69	Robert	Stevens	US	500	Once a Year

Compilation

```
data work.bonus;  
  set orion.sales;  
  if Country='US' then do;  
    Bonus=500;  
    Freq='Once a Year';  
  end;  
  else if Country='AU' then do;  
    Bonus=300;  
    Freq='Twice a Year';  
  end;  
run;
```

PDV

Employee_ID	First_Name	...	Hire_Date
N 8	\$ 12		N 8

Compilation

```
data work.bonus;  
  set orion.sales;  
  if Country='US' then do;  
    Bonus=500;  
    Freq='Once a Year';  
  end;  
  else if Country='AU' then do;  
    Bonus=300;  
    Freq='Twice a Year';  
  end;  
run;
```

PDV

Employee_ID	First_Name
N 8	\$ 12

...

Hire_Date	Bonus
N 8	N 8

Compilation

```
data work.bonus;  
  set orion.sales;  
  if Country='US' then do;  
    Bonus=500;  
    Freq='Once a Year';  
  end;  
  else if (11 characters) then do;  
    Bonus=500;  
    Freq='Twice a Year';  
  end;  
run;
```

PDV

Employee_ID	First_Name
N 8	\$ 12

...

Hire_Date	Bonus	Freq
N 8	N 8	\$ 11

Poll

Quiz



9.06 Quiz

How would you prevent **Freq** from being truncated?

9.06 Quiz – Correct Answer

How would you prevent **Freq** from being truncated?

Possible solutions:

- Pad the first occurrence of the **Freq** value with blanks to be the length of the longest possible value.
- Switch conditional statements to place the longest value of **Freq** in the first conditional statement.
- Add a **LENGTH** statement to declare the byte size of the variable up front.

The LENGTH Statement (Review)

The *LENGTH* statement defines the length of a variable explicitly.

General form of the LENGTH statement:

```
LENGTH variable(s) $ length;
```

Example:

```
length First_Name Last_Name $ 12  
        Gender $ 1;
```

Business Scenario

Set the length of the variable **Freq** to avoid truncation.

```
data work.bonus;  
  set orion.sales;  
  length Freq $ 12;  
  if Country='US' then do;  
    Bonus=500;  
    Freq='Once a Year';  
  end;  
  else if Country='AU' then do;  
    Bonus=300;  
    Freq='Twice a Year';  
  end;  
run;
```

Business Scenario

```
proc print data=work.bonus;  
  var First Name Last Name  
      Country Bonus Freq;  
run;
```

Partial PROC PRINT Output

Obs	First_Name	Last_Name	Country	Bonus	Freq
60	Billy	Plested	AU	300	Twice a Year
61	Matsuoka	Wills	AU	300	Twice a Year
62	Vino	George	AU	300	Twice a Year
63	Meera	Body	AU	300	Twice a Year
64	Harry	Highpoint	US	500	Once a Year
65	Julienne	Magolan	US	500	Once a Year
66	Scott	Desanctis	US	500	Once a Year
67	Cherda	Ridley	US	500	Once a Year
68	Priscilla	Farren	US	500	Once a Year
69	Robert	Stevens	US	500	Once a Year

ELSE Statements

The conditional clause does not have to be in an ELSE statement. (An ELSE does not require a second IF.)

```
data work.bonus;  
  set orion.sales;  
  length Freq $ 12;  
  if Country='US' then do;  
    Bonus=500;  
    Freq='Once a Year';  
  end;  
  else do;  
    Bonus=300;  
    Freq='Twice a Year';  
  end;  
run;
```



All observations not equal to US execute the statements in the second DO group.



Question & Answer

Chapter 9: Manipulating Data



9.1 Creating Variables

9.2 Creating Variables Conditionally

9.3 Subsetting Observations

Objectives

- Subset observations by using the WHERE statement.
- Subset observations by using the subsetting IF statement.
- Subset observations by using the IF-THEN DELETE statement. (Self-Study)

Business Scenario

A new SAS data set named **Work.december** needs to be created by reading the **orion.sales** data set.

Work.december must include the following new variables:

- **Bonus**, which is equal to a constant 500.
- **Compensation**, which is the combination of the employee's salary and bonus.
- **BonusMonth**, which is equal to the month the employee was hired.

Work.december must include only the employees from Australia who have a bonus month in December.

The WHERE Statement (Review)

The *WHERE* statement subsets observations that meet a particular condition.

General form of the WHERE statement:

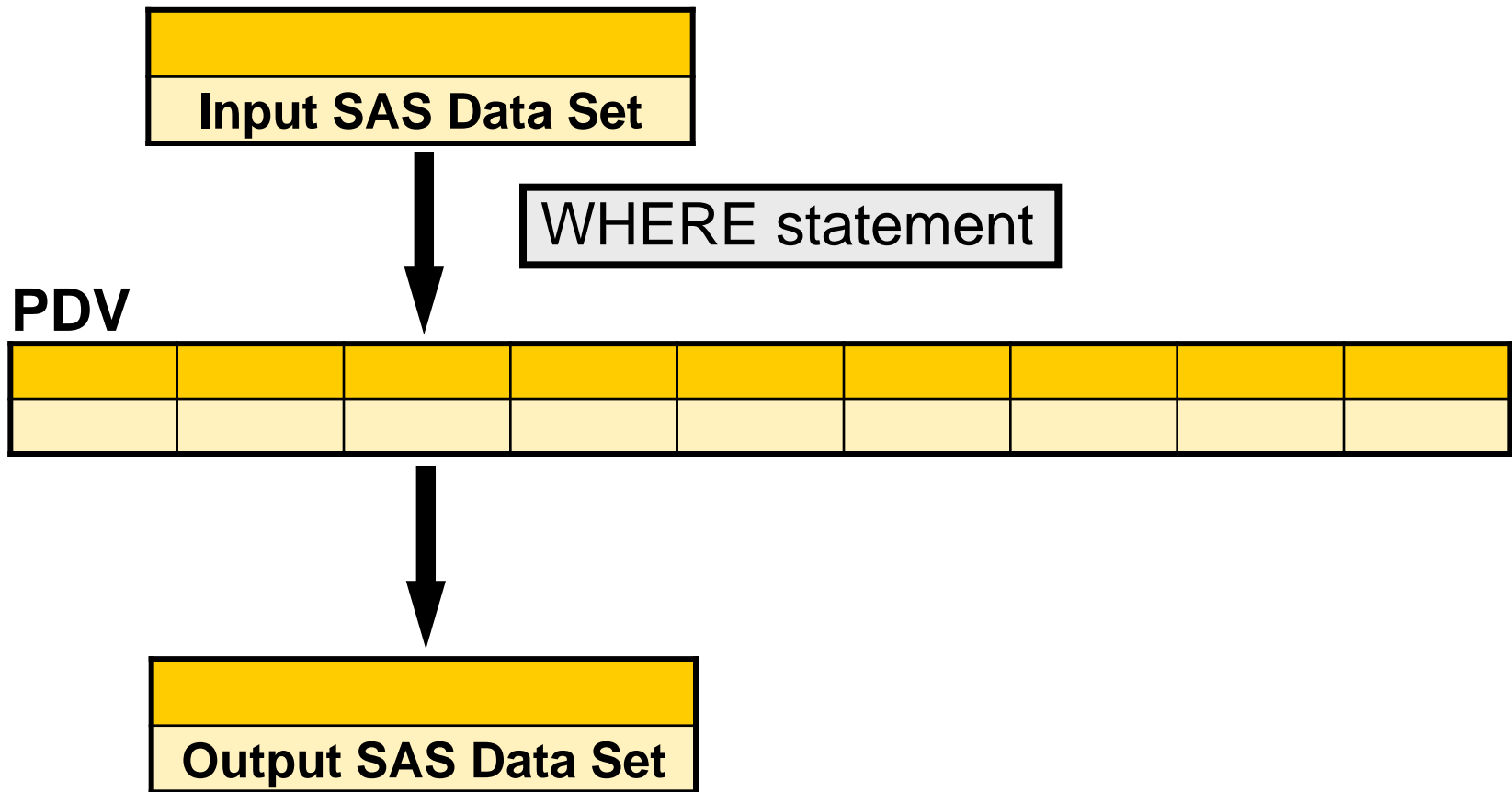
```
WHERE where-expression;
```

The *where-expression* is a sequence of operands and operators that form a set of instructions that define a condition for selecting observations.

- Operands include constants and variables.
- Operators are symbols that request a comparison, arithmetic calculation, or logical operation.

Processing the WHERE Statement

The WHERE statement selects observations **before** they are brought into the program data vector.



Poll

Quiz



9.07 Quiz

Why does the WHERE statement not work in this DATA step?

```
data work.december;  
    set orion.sales;  
    BonusMonth=month(Hire_Date) ;  
    Bonus=500;  
    Compensation=sum(Salary,Bonus) ;  
    where Country='AU' and BonusMonth=12;  
run;
```

9.07 Quiz – Correct Answer

Why does the WHERE statement not work in this DATA step?

```
data work.december;  
    set orion.sales;  
    BonusMonth=month(Hire_Date) ;  
    Bonus=500 ;  
    Compensation=sum(Salary,Bonus) ;  
    where Country='AU' and BonusMonth=12 ;  
run ;
```

The WHERE statement can only subset variables that are coming from an existing data set.

ERROR: Variable BonusMonth is not on file ORION.SALES.

The Subsetting IF Statement

The *subsetting IF statement* continues processing only those observations that meet the condition.

General form of the subsetting IF statement:

IF *expression*;

The *expression* is a sequence of operands and operators that form a set of instructions that define a condition for selecting observations.

- Operands include constants and variables.
- Operators are symbols that request a comparison, arithmetic calculation, or logical operation.

The Subsetting IF Statement

Examples:

```
if Salary > 50000;
```

```
if Last_Name='Smith' and First_Name='Joe';
```

```
if Country not in ('GB', 'FR', 'NL');
```

```
if Hire_Date = '15APR2008'd;
```

```
if BirthMonth = 5 or BirthMonth = 6;
```

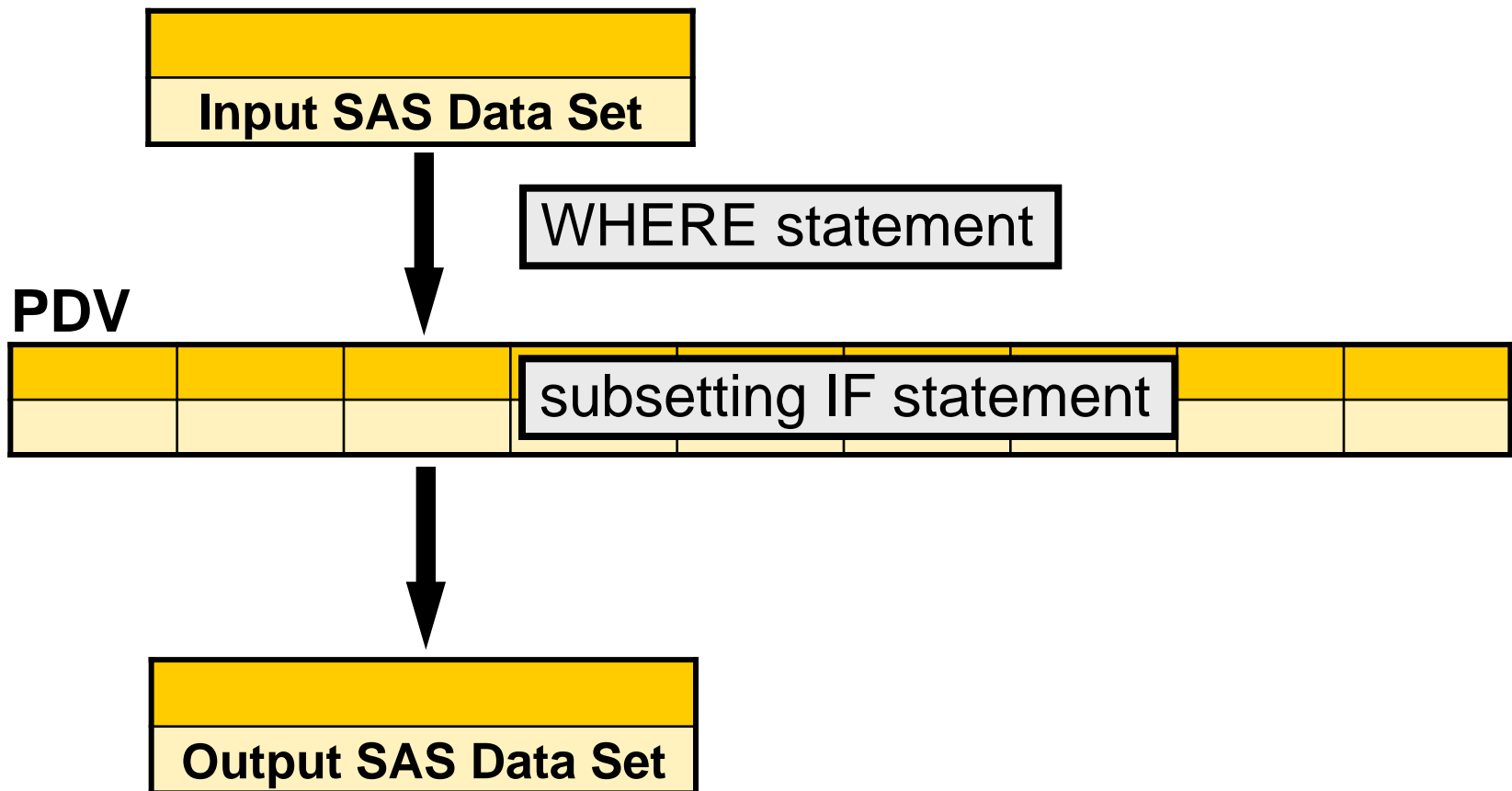
```
if upcase(Gender)='M';
```

```
if 40000 <= Compensation <= 80000;
```

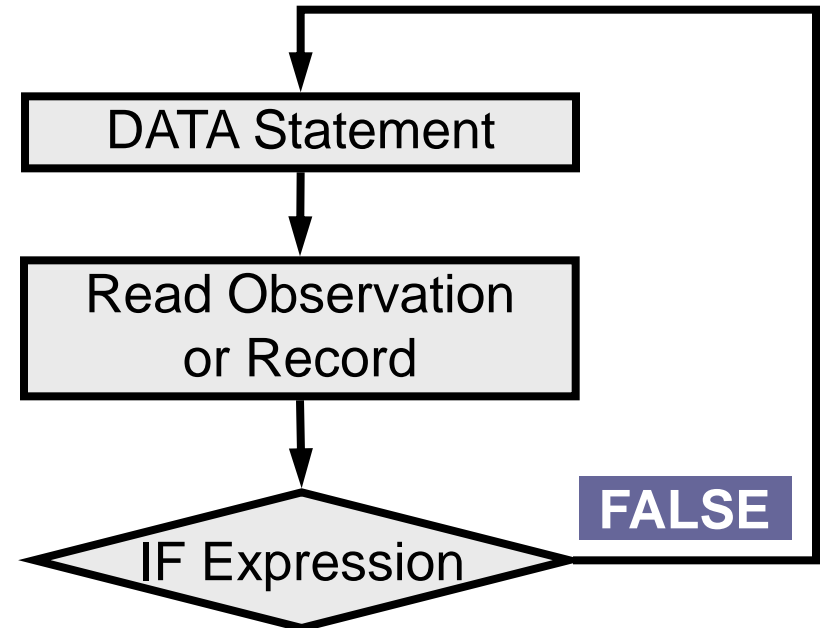
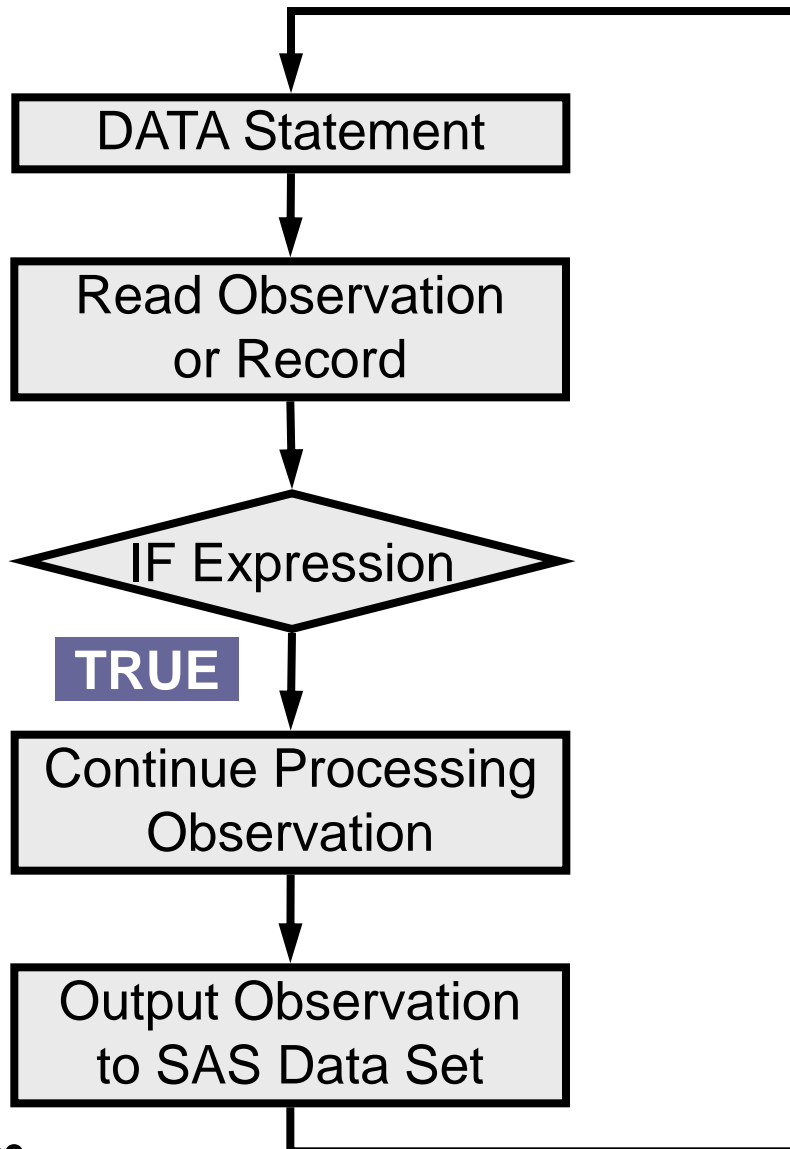
```
if sum(Salary,Bonus) < 43000;
```

Processing the Subsetting IF Statement

The subsetting IF statement determines if observations continue being processed in the program data vector.



Processing the Subsetting IF Statement



A false IF expression causes the contents of the PDV to not output.

Business Scenario

Include only the employees from Australia who have a bonus month in December.

```
data work.december;  
  set orion.sales;  
  where Country='AU';  
  BonusMonth=month(Hire_Date);  
  if BonusMonth=12;  
  Bonus=500;  
  Compensation=sum(Salary,Bonus);  
run;
```

Partial SAS Log

```
NOTE: There were 63 observations read from the data set ORION.SALES.  
      WHERE Country='AU';  
NOTE: The data set WORK.DECEMBER has 3 observations and 12 variables.
```

Poll

Quiz



9.08 Quiz

Could you write only an IF statement?

- ☐ Yes
- ☐ No

```
data work.december;  
  set orion.sales;  
  where Country='AU';  
  BonusMonth=month(Hire_Date);  
  if BonusMonth=12;  
    Bonus=500;  
  Compensation  
run;
```

```
data work.december;  
  set orion.sales;  
  BonusMonth=month(Hire_Date);  
  if BonusMonth=12 and Country='AU';  
    Bonus=500;  
  Compensation=sum(Salary,Bonus);  
run;
```

9.08 Quiz – Correct Answer

Could you write only an IF statement?

- ☒ Yes
- ☐ No

Yes, but the program using both the WHERE and IF statements is more efficient.

Both methods create a data set with three observations. The program using both statements reads 63 observations into the PDV. The program using only the IF statement reads 165 observations into the PDV.

WHERE Statement versus Subsetting IF Statement

Step and Usage	WHERE	IF
PROC step	Yes	No
DATA step (source of variable)		
INPUT statement	No	Yes
assignment statement	No	Yes
SET statement (single data set)	Yes	Yes
SET/MERGE statement (multiple data sets)		
Variable in ALL data sets	Yes	Yes
Variable not in ALL data sets	No	Yes



Question & Answer

The IF-THEN DELETE Statement (Self-Study)

An alternative to the subsetting IF statement is the DELETE statement in an IF-THEN statement.

General form of the IF-THEN DELETE statement:

IF *expression* THEN DELETE;

The *DELETE statement* stops processing the current observation.

The IF-THEN DELETE Statement (Self-Study)

```
data work.december;  
  set orion.sales;  
  where Country='AU';  
  BonusMonth=month(Hire_Date);  
  if BonusMonth ne 12 then delete;  
  Bonus=500;  
  Compensation=sum(Salary,Bonus);  
run;
```

equivalent

```
data work.december;  
  set orion.sales;  
  where Country='AU';  
  BonusMonth=month(Hire_Date);  
  if BonusMonth=12;  
  Bonus=500;  
  Compensation=sum(Salary,Bonus);  
run;
```

Chapter 2: Controlling Input and Output



2.1 Outputting Multiple Observations

2.2 Writing to Multiple SAS Data Sets

2.3 Selecting Variables and Observations

Chapter 2: Controlling Input and Output

2.1 Outputting Multiple Observations

2.2 Writing to Multiple SAS Data Sets

2.3 Selecting Variables and Observations

Objectives

- Explicitly control the output of multiple observations to a SAS data set.

Business Scenario – A Forecasting Application

The growth rate of six departments at Orion Star is stored in the **Increase** variable in the data set **orion.growth**. If each department grows at its predicted rate for the next two years, how many employees will be in each department at the end of each year?

Listing of **orion.growth**

Department	Total_ Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS	25	0.10
Marketing	20	0.20
Sales	201	0.30
Sales Management	11	0.10

A Forecasting Application

The output SAS data set, **forecast**, should contain 12 observations. Two observations are written for each observation read.

Partial Listing of **forecast**

Department	Total_ Employees	Increase	Year
Administration	42.500	0.25	1
Administration	53.125	0.25	2
Engineering	11.700	0.30	1
Engineering	15.210	0.30	2
IS	27.500	0.10	1
IS	30.250	0.10	2

Poll

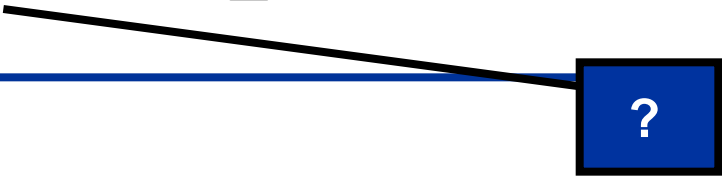
Quiz



2.01 Quiz

Which of the following occur at the end of a DATA step iteration?

```
data forecast;  
    set orion.growth;  
    total_employees=  
        total_employees * (1+increase);  
run;
```



- a. Reinitialize the PDV.
- b. Implicit OUTPUT; implicit RETURN.
- c. Read the next observation.

2.01 Quiz – Correct Answer

Which of the following occur at the end of a DATA step iteration?

```
data forecast;  
    set orion.growth;  
    total_employees=  
        total_employees * (1+increase);  
run;
```

**Implicit OUTPUT;
Implicit RETURN;**

By default, every DATA step performs an implicit OUTPUT and implicit RETURN at the end of each iteration.

Explicit Output

The explicit OUTPUT statement writes the contents of the program data vector (PDV) to the data set or data sets being created. The presence of an explicit OUTPUT statement overrides implicit output.

```
data forecast;  
  set orion.growth;  
  Year=1;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
  Year=2;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
run;
```

No Implicit OUTPUT;

Compilation

```
data forecast;  
  set orion.growth;  
  Year=1;  
  Total_Employees=Total_Employees*(1+Increase) ;  
  output;  
  Year=2;  
  Total_Employees=Total_Employees*(1+Increase) ;  
  output;  
run;
```

PDV – Program Data Vector

Department \$ 20	Total_ Employees N 8	Increase N 8

Compilation

```
data forecast;  
  set orion.growth;  
  Year=1;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
  Year=2;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
run;
```

PDV

Department \$ 20	Total_ Employees N 8	Increase N 8	Year N 8

Compilation

```
data forecast;  
  set orion.growth;  
  Year=1;  
  Total_Employees=Total_Employees*(1+Increase) ;  
  output;  
  Year=2;  
  Total_Employees=Total_Employees*(1+Increase) ;  
  output;  
run;
```

Write descriptor portion of output data set

PDV

Department \$ 20	Total_ Employees N 8	Increase N 8	Year N 8

work.forecast

Department \$ 20	Total_ Employees N 8	Increase N 8	Year N 8

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```
data forecast;
```

```
set orion.growth;
```

```
Year=1;
```

```
Total_Employees=Total_Employees*(1+Increase);
```

```
output;
```

```
Year=2;
```

```
Total_Employees=Total_Employees*(1+Increase);
```

```
output;
```

```
run;
```

Initialize PDV

PDV

Department	Total_Employees	Increase	Year
\$ 20	N 8	N 8	N 8
	.	.	.

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```
data forecast;
  set orion.growth;
  Year=1;
  Total_Employees=Total_Employees*(1+Increase);
  output;
  Year=2;
  Total_Employees=Total_Employees*(1+Increase);
  output;
run;
```

PDV

Department	Total_ Employees	Increase	Year
\$ 20	N 8	N 8	N 8
Administration	34	0.25	.

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```
data forecast;  
  set orion.growth;  
  Year=1;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
  Year=2;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
run;
```

PDV

Department	Total_ Employees	Increase	Year
\$ 20	N 8	N 8	N 8
Administration	34	0.25	1

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```
data forecast;  
  set orion.growth;  
  Year=1;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
  Year=2;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
run;
```

$34 * (1 + 0.25)$

PDV

Department	Total_Employees	Increase	Year
\$ 20	N 8	N 8	N 8
Administration	42.5	0.25	1

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```
data forecast;
  set orion.growth;
  Year=1;
  Total_Employees=Total_Employees*(1+Increase);
  output;
  Year=2;
  Total_Employees=Total_Employees*(1+Increase);
  output;
run;
```

Output current observation

PDV

Department	Total_Employees	Increase	Year
\$ 20	N 8	N 8	N 8
Administration	42.5	0.25	1

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```
data forecast;  
  set orion.growth;  
  Year=1;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
  Year=2;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
run;
```

PDV

Department	Total_ Employees	Increase	Year
\$ 20	N 8	N 8	N 8
Administration	42.5	0.25	2

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```
data forecast;  
  set orion.growth;  
  Year=1;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
  Year=2;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
run;
```

$$42.5 * (1 + 0.25)$$

PDV

Department	Total_Employees	Increase	Year
\$ 20	N 8	N 8	N 8
Administration	53.125	0.25	2

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```
data forecast;
  set orion.growth;
  Year=1;
  Total_Employees=Total_Employees*(1+Increase);
  output;
  Year=2;
  Total_Employees=Total_Employees*(1+Increase);
  output;
run;
```

Output current observation

PDV

Department	Total_Employees	Increase	Year
\$ 20	N 8	N 8	N 8
Administration	53.125	0.25	2

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```
data forecast;  
  set orion.growth;  
  Year=1;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
  Year=2;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
run;
```

No Implicit OUTPUT;

PDV

Department	Total_ Employees	Increase	Year
\$ 20	N 8	N 8	N 8
Administration	53.125	0.25	2

Output: A Forecasting Application

The **forecast** data set contains two observations after the first iteration of the DATA step.

work.forecast

Department	Total_ Employees	Increase	Year
Administration	42.500	0.25	1
Administration	53.125	0.25	2

Poll 

Quiz

Setup for the Poll

Prior to the second iteration of the DATA step, some variables in the program data vector will be reinitialized.

```
data forecast;  
    set orion.growth;  
    Year=1;  
    Total_Employees=Total_Employees*(1+Increase);  
    output;  
    Year=2;  
    Total_Employees=Total_Employees*(1+Increase);  
    output;  
run;
```

PDV

Department \$ 20	Total_ Employees N 8	Increase N 8	Year N 8
Administration	53.125	0.25	2

2.02 Multiple Answer Poll

Which variable(s) will be reinitialized?

- a. **Department**
- b. **Total_Employees**
- c. **Increase**
- d. **Year**

2.02 Multiple Answer Poll – Correct Answers

Which variable(s) will be reinitialized?

- a. **Department**
- b. **Total_Employees**
- c. **Increase**
- ☒ d. **Year**

Variables created by INPUT and assignment statements are reinitialized. Variables read with a SET statement are not.

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```
data forecast;
```

```
set orion.growth;
```

```
Year=1;
```

```
Total_Employees=Total_Employees*(1+Increase);
```

```
output;
```

```
Year=2;
```

```
Total_Employees=Total_Employees*(1+Increase);
```

```
output;
```

```
run;
```

Reinitialize PDV

PDV

Department	Total_Employees	Increase	Year
\$ 20	N 8	N 8	N 8
Administration	53.125	0.25	.

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```
data forecast;  
  set orion.growth;  
  Year=1;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
  Year=2;  
  Total_Employees=Total_Employees*(1+Increase);  
  output;  
run;
```

PDV

Department	Total_ Employees	Increase	Year
\$ 20	N 8	N 8	N 8
Engineering	9	0.30	.

Execution: Explicit Output

orion.growth

Department	Total_Employees	Increase
Administration	34	0.25
Engineering	9	0.30
IS		0

```

data forecast;
  set orion.growth;
  Year=1;
  Total_Emp = Total_Employees * (1+Increase);
  output;
  Year=2;
  Total_Employees=Total_Employees * (1+Increase);
  output;
run;
  
```

Continue until EOF

PDV

Department	Total_Employees	Increase	Year
\$ 20	N 8	N 8	N 8
Engineering	9	0.30	1

Check the Results

Partial SAS Log

NOTE: There were 6 observations read from the data set
ORION.GROWTH.

NOTE: The data set WORK.FORECAST has 12 observations
and 4 variables.

Partial PROC PRINT Output

Department	Total_ Employees	Year
Administration	42.500	1
Administration	53.125	2
Engineering	11.700	1
Engineering	15.210	2
IS	27.500	1
IS	30.250	2
Marketing	24.000	1
Marketing	28.800	2

Poll

Quiz



2.03 Quiz

Open and submit **p202a01**. Modify the DATA step to write only one observation per department. Show the number of employees after two years.

Desired Results

Department	Total_ Employees	Year
Administration	53.125	2
Engineering	15.210	2
IS	30.250	2
Marketing	28.800	2
Sales	339.690	2
Sales Management	13.310	2

2.03 Quiz – Correct Answer

There are several ways to modify the DATA step.
Here is one solution:

```
data forecast;  
    set orion.growth;  
    Year=1;  
    Total_Employees=Total_Employees*(1+Increase) ;  
    Year=2;  
    Total_Employees=Total_Employees*(1+Increase) ;  
    output;  
run;
```

Chapter 2: Controlling Input and Output



2.1 Outputting Multiple Observations

2.2 Writing to Multiple SAS Data Sets

2.3 Selecting Variables and Observations

Objectives

- Create multiple SAS data sets in a single DATA step.
- Use conditional processing to control the data set(s) to which an observation is written.

Business Scenario

Use the **orion.employee_addresses** data set as input to create three new data sets: **usa**, **australia**, and **other**.

- The **usa** data set will contain observations with a **Country** value of **US**.
- The **australia** data set will contain observations with a **Country** value of **AU**.
- Observations with any other **Country** value will be written to the **other** data set.

Browse the Input Data Set

```
proc print data=orion.employee_addresses;  
    var Employee_Name City Country;  
run;
```

Partial PROC PRINT Output

Obs	Employee_Name	City	Country
1	Abbott, Ray	Miami-Dade	US
2	Aisbitt, Sandy	Melbourne	AU
3	Akinfolarin, Tameaka	Philadelphia	US
4	Amos, Salley	San Diego	US
5	Anger, Rose	Philadelphia	US
6	Anstey, David	Miami-Dade	US
7	Antonini, Doris	Miami-Dade	US
8	Apr, Nishan	San Diego	US
9	Ardskin, Elizabeth	Miami-Dade	US
10	Areu, Jeryl	Miami-Dade	US
11	Arizmendi, Gilbert	San Diego	US
12	Armant, Debra	San Diego	US

Creating Multiple DATA Sets

Multiple data sets can be created in a DATA step by listing the names of the output data sets in the DATA statement.

```
DATA <SAS-data-set-1 ... SAS-data-set-n>;
```

You can direct output to a specific data set or data sets by listing the data set names in the OUTPUT statement.

```
OUTPUT <SAS-data-set-1 ... SAS-data-set-n>;
```

An OUTPUT statement without arguments writes to every SAS data set listed in the DATA statement.

Creating Multiple SAS Data Sets

Create three new data sets: **usa**, **australia**, and **other**.

```
data usa australia other;  
  set orion.employee_addresses;  
  if Country='AU' then output australia;  
  else if Country='US' then output usa;  
  else output other;  
run;
```

Check the SAS Log

Three data sets were created. The log shows that US was the most frequently occurring value.

Partial SAS Log

```
NOTE: There were 424 observations read from the data set  
      ORION.EMPLOYEE_ADDRESSES.  
NOTE: The data set WORK.USA has 311 observations and 9  
      variables.  
NOTE: The data set WORK.AUSTRALIA has 105 observations and  
      9 variables.  
NOTE: The data set WORK.OTHER has 8 observations and 9  
      variables.
```

Efficient Conditional Processing

It is more efficient to check values in order of decreasing frequency.

```
data usa australia other;  
  set orion.employee_addresses;  
  if Country='US' then output usa;  
  else if Country='AU' then output australia;  
  else output other;  
run;
```

NOTE: There were 424 observations read from the data set ORION.EMPLOYEE_ADDRESSES.

NOTE: The data set WORK.USA has 311 observations and 9 variables.

NOTE: The data set WORK.AUSTRALIA has 105 observations and 9 variables.

NOTE: The data set WORK.OTHER has 8 observations and 9 variables.

Poll

Quiz



2.04 Quiz

Consider the results of the previous DATA step.
Can all three data sets be printed with a single
PRINT procedure?

Partial SAS Log

```
NOTE: There were 424 observations read from the data set  
      ORION.EMPLOYEE_ADDRESSES.  
NOTE: The data set WORK.USA has 311 observations and 9  
      variables.  
NOTE: The data set WORK.AUSTRALIA has 105 observations and  
      9 variables.  
NOTE: The data set WORK.OTHER has 8 observations and 9  
      variables.
```

2.04 Quiz – Correct Answer

Consider the results of the previous DATA step.
Can all three data sets be printed with a single PRINT procedure?

Partial SAS Log

```
NOTE: There were 424 observations read from the data set  
      ORION.EMPLOYEE_ADDRESSES.  
NOTE: The data set WORK.USA has 311 observations and 9  
      variables.  
NOTE: The data set WORK.AUSTRALIA has 105 observations and  
      9 variables.  
NOTE: The data set WORK.OTHER has 8 observations and 9  
      variables.
```

No, a separate PRINT procedure is needed for each data set.

Displaying Multiple SAS Data Sets

The PRINT procedure can only print one data set. A separate PROC PRINT step is required for each data set.

```
title 'Employees in the United States';  
proc print data=usa;  
run;
```

```
title 'Employees in Australia';  
proc print data=australia;  
run;
```

```
title 'Non US and AU Employees';  
proc print data=other;  
run;
```

```
title;
```

Using a SELECT Group

An alternate form of conditional execution uses a SELECT group.

```
SELECT <(select-expression)>;  
    WHEN-1 (value-1 <...,value-n>  
        statement,  
    <...WHEN-n (value-1 <...,value-n>  
        statement,>  
    <OTHERWISE statement,>  
END;
```

The *select-expression* specifies any SAS expression that evaluates to a single value.

 Often a variable name is used as the *select-expression*.

Using a SELECT Group

The previous task can be rewritten using a SELECT group:

```
data usa australia other;  
  set orion.employee_addresses;  
  select (Country);  
    when ('US') output usa;  
    when ('AU') output australia;  
    otherwise output other;  
  end;  
run;
```

The SELECT statement processes the WHEN statements from top to bottom, so it is more efficient to check the values in order of decreasing frequency.

Check the SAS Log

Results using SELECT are the same as IF-THEN/ELSE results.

Partial SAS Log

NOTE: There were 424 observations read from the data set
ORION.EMPLOYEE_ADDRESSES.

NOTE: The data set WORK.USA has 311 observations and 9
variables.

NOTE: The data set WORK.AUSTRALIA has 105 observations and 9
variables.

NOTE: The data set WORK.OTHER has 8 observations and 9
variables.

The OTHERWISE Statement

The OTHERWISE statement is optional, but omitting it results in an error when all WHEN conditions are false.

```
SELECT <(select-expression)>;  
    WHEN-1 (value-1 <...,value-n>  
        statement,  
    <...WHEN-n (value-1 <...,value-n>  
        statement,>  
    <OTHERWISE statement,>  
END;
```



Use the OTHERWISE statement followed by a null statement to prevent SAS from issuing an error message.

```
otherwise;
```

Poll

Quiz



2.05 Quiz

Open the file **p202a03** and submit it. View the log, identify and correct the problem, and resubmit the program.

```
data usa australia;  
  set orion.employee_addresses;  
  select (Country);  
    when ('US') output usa;  
    when ('AU') output australia;  
end;  
run;
```

2.05 Quiz – Correct Answer

Open the file **p202a03** and submit it. View the log, identify and correct the problem, and resubmit the program.

```
150 data usa australia;
151     set orion.employee_addresses;
152     select (Country);
153         when ('US') output usa;
154         when ('AU') output australia;
155     end;
156 run;
```

ERROR: Unsatisfied WHEN clause and no OTHERWISE clause at line 155 column 4.

```
data usa australia;
    set orion.employee_addresses;
    select (Country);
        when ('US') output usa;
        when ('AU') output australia;
        otherwise;
    end;
run;
```

**An OTHERWISE
statement is needed.**

Test for Multiple Values in a WHEN Statement

Multiple values can be listed in the WHEN expression.

```
data usa australia other;
  set orion.employee_addresses;
  select (Country);
    when ('US','us') output usa;
    when ('AU','au') output australia;
    otherwise output other;
  end;
run;
```

Partial SAS Log

```
NOTE: There were 424 observations read from the data set
      ORION.EMPLOYEE_ADDRESSES.
NOTE: The data set WORK.USA has 316 observations and 9 variables.
NOTE: The data set WORK.AUSTRALIA has 108 observations and 9
      variables.
NOTE: The data set WORK.OTHER has 0 observations and 9 variables.
```

Using Functions in a Select Expression

An alternate solution uses the UPCASE function.

```
data usa australia other;  
  set orion.employee_addresses;  
  select (upcase(Country));  
    when ('US') output usa;  
    when ('AU') output australia;  
    otherwise output other;  
end;  
run;
```

Partial SAS Log

```
NOTE: There were 424 observations read from the data set  
      ORION.EMPLOYEE_ADDRESSES.  
NOTE: The data set WORK.USA has 316 observations and 9 variables.  
NOTE: The data set WORK.AUSTRALIA has 108 observations and 9  
      variables.  
NOTE: The data set WORK.OTHER has 0 observations and 9 variables.
```

Using DO-END in a SELECT Group

Use DO and END statements to execute multiple statements when an expression is true.

```
data usa australia other;
  set orion.employee_addresses;
  select (upcase(country));
    when ('US') do;
      Benefits=1;
      output usa;
    end;
    when ('AU') do;
      Benefits=2;
      output australia;
    end;
    otherwise do;
      Benefits=0;
      output other;
    end;
  end;
run;
```

Omitting the Select Expression

The *select-expression* can be omitted in a SELECT group:

```
SELECT;  
    WHEN (expression-1)  
        statement;  
    <...WHEN (expression-n)  
        statement; >  
    <OTHERWISE statement; >  
END;
```

Each WHEN expression evaluates to true or false.

- If true, the associated statement(s) is executed.
- If false, SAS proceeds to the next WHEN statement.
- If all WHEN expressions are false, then the statement(s) following the OTHERWISE statement executes.

Omitting the Select Expression

This version of the current example omits the SELECT expression:

```
data usa australia other;  
  set orion.employee_addresses;  
  select;  
    when (country='US') output usa;  
    when (country='AU') output australia;  
    otherwise output other;  
end;  
run;
```

Partial SAS Log

```
NOTE: There were 424 observations read from the data set  
      ORION.EMPLOYEE_ADDRESSES.  
NOTE: The data set WORK.USA has 311 observations and 9 variables.  
NOTE: The data set WORK.AUSTRALIA has 105 observations and 9  
      variables.  
NOTE: The data set WORK.OTHER has 8 observations and 9 variables.
```