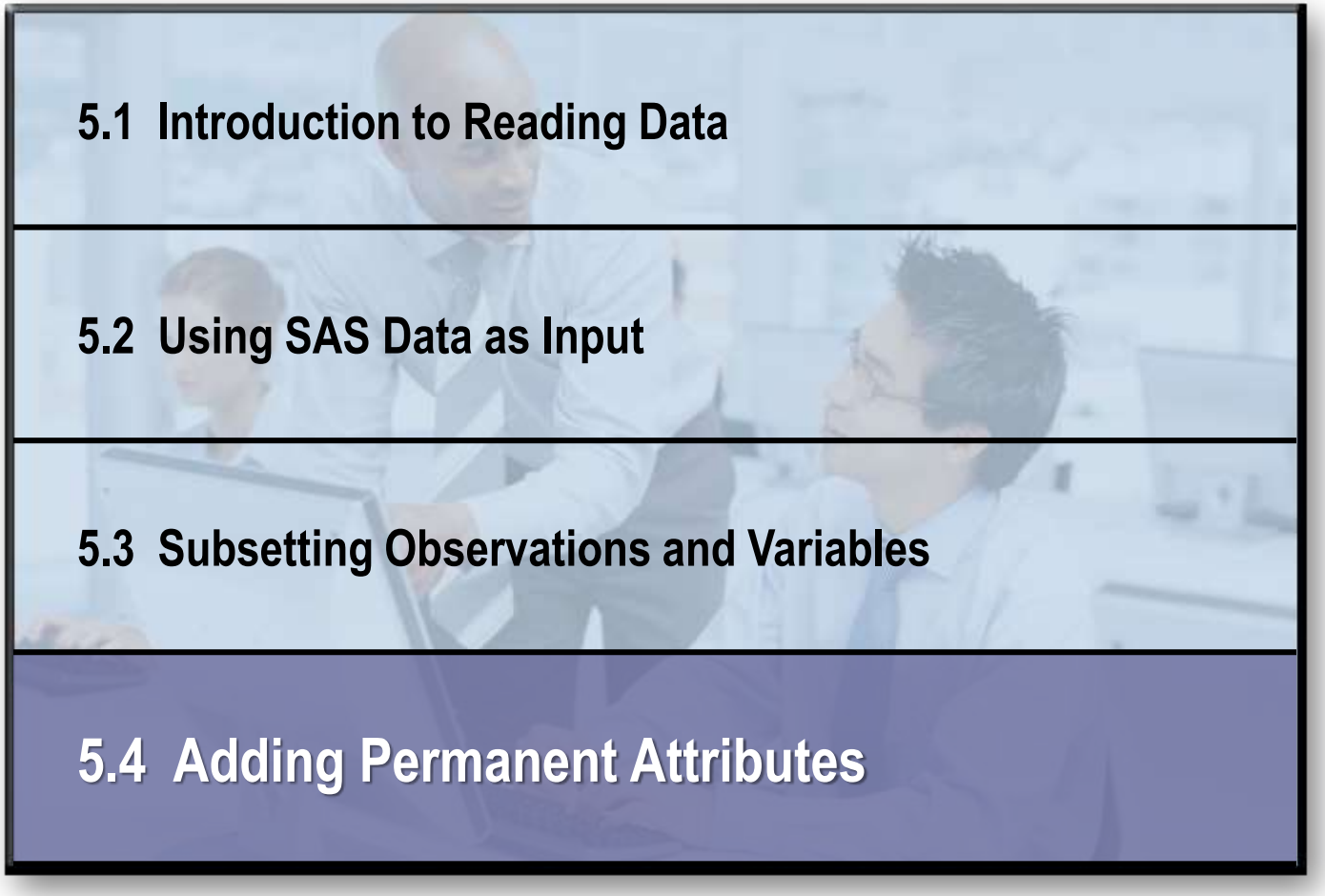
STAT604

Lesson SAS 04

THE POWER TO KNOW®

Chapter 5: Reading SAS Data Sets

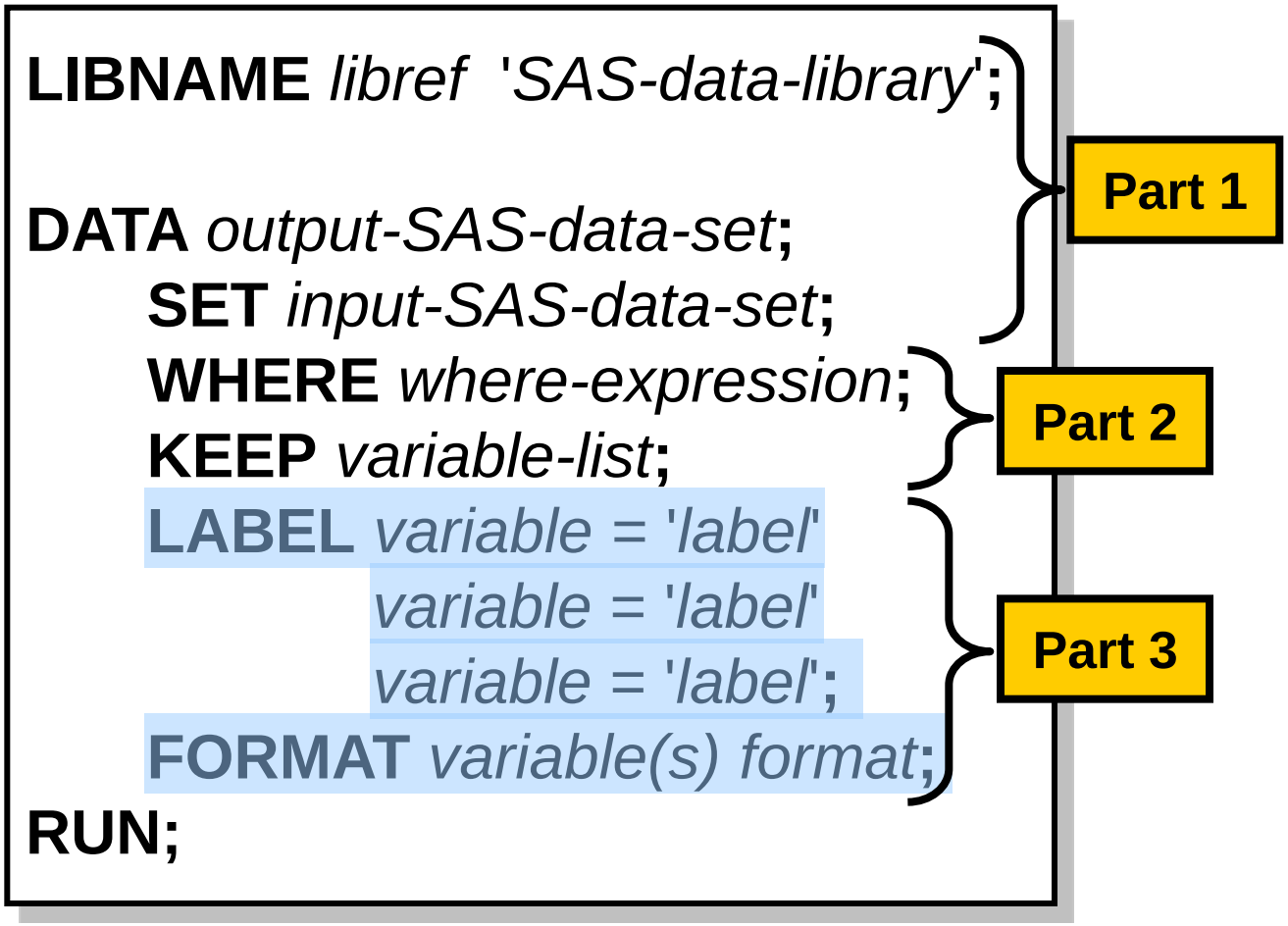


Objectives

- Add labels to the descriptor portion of a SAS data set by using the LABEL statement.
- Add formats to the descriptor portion of a SAS data set by using the FORMAT statement.

Business Scenario Syntax

Use the following statements to complete the scenario:



Adding Permanent Attributes

The descriptor portion of the SAS data set stores variable attributes including the name, type (character or numeric), and length of the variable.

Labels and formats can also be stored in the descriptor portion.

Partial PROC CONTENTS Output

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Label
1	First_Name	Char	12		
5	Hire_Date	Num	8	DDMMYY10.	Date Hired
4	Job_Title	Char	25		Sales Title
2	Last_Name	Char	18		
3	Salary	Num	8	COMMAX8.	

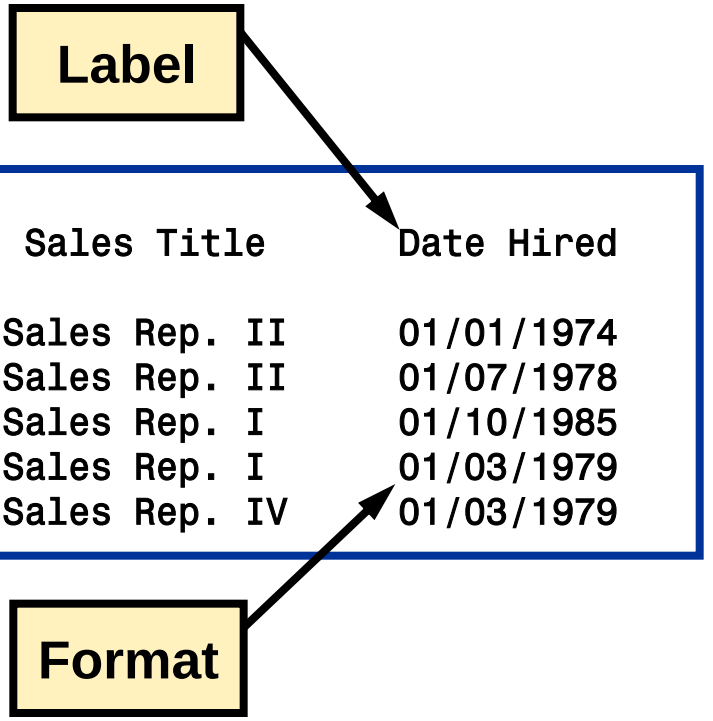
Adding Permanent Attributes

When displaying reports,

- a *label* changes the appearance of a variable name
- a *format* changes the appearance of variable value.

Partial PROC PRINT Output

Obs	First_ Name	Last_Name	Salary	Sales Title	Date Hired
1	Irenie	Elvish	26.600	Sales Rep. II	01/01/1974
2	Christina	Ngan	27.475	Sales Rep. II	01/07/1978
3	Kimiko	Hotstone	26.190	Sales Rep. I	01/10/1985
4	Lucian	Daymond	26.480	Sales Rep. I	01/03/1979
5	Fong	Hofmeister	32.040	Sales Rep. IV	01/03/1979



The LABEL Statement

The *LABEL statement* assigns descriptive labels to variable names.

General form of the LABEL statement:

LABEL *variable* = '*label*'
variable = '*label*'
variable = '*label*';

- A label can have as many as 256 characters.
- Any number of variables can be associated with labels in a single LABEL statement.
- Using a LABEL statement in a DATA step permanently associates labels with variables by storing the label in the descriptor portion of the SAS data set.

Business Scenario Part 3

Include labels in the descriptor portion of **Work.subset1**.

```
data work.subset1;
   set orion.sales;
   where Country='AU' and
         Job_Title contains 'Rep';
   keep First_Name Last_Name Salary
        Job_Title Hire_Date;
   label Job_Title='Sales Title'
         Hire_Date='Date Hired';
run;
```

Business Scenario Part 3

```
proc contents data=work.subset1;
run;
```

Partial PROC CONTENTS Output

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Label
1	First_Name	Char	12	
5	Hire_Date	Num	8	Date Hired
4	Job_Title	Char	25	Sales Title
2	Last_Name	Char	18	
3	Salary	Num	8	

Business Scenario Part 3

In order to use labels in the PRINT procedure, a LABEL option needs to be added to the PROC PRINT statement.

```
proc print data=work.subset1 label;
run;
```

Partial PROC PRINT Output

Obs	First_ Name	Last_Name	Salary	Sales Title	Date Hired
1	Irenie	Elvish	26600	Sales Rep. II	5114
2	Christina	Ngan	27475	Sales Rep. II	6756
3	Kimiko	Hotstone	26190	Sales Rep. I	9405
4	Lucian	Daymond	26480	Sales Rep. I	6999
5	Fong	Hofmeister	32040	Sales Rep. IV	6999
6	Satyakam	Denny	26780	Sales Rep. II	17014
7	Sharryn	Clarkson	28100	Sales Rep. II	14184
8	Monica	Kletschkus	30890	Sales Rep. IV	17106
9	Alvin	Roebuck	30070	Sales Rep. III	9405
10	Kevin	Lyon	26955	Sales Rep. I	16922

p105d04

The FORMAT Statement

The *FORMAT statement* assigns formats to variable values.

General form of the FORMAT statement:

FORMAT *variable(s) format*;

- A *format* is an instruction that SAS uses to write data values.
- Using a FORMAT statement in a DATA step permanently associates formats with variables by storing the format in the descriptor portion of the SAS data set.

SAS Formats

SAS formats have the following form:

<$>*format*<*w*>.<*d*>

$	indicates a character format.
format	names the SAS format or user-defined format.
w	specifies the total format width including decimal places and special characters.
.	is a required delimiter.
d	specifies the number of decimal places in numeric formats.

SAS Formats

Selected SAS formats:

Format	Definition
$*w*.	writes standard character data.
w.*d*	writes numeric values with a comma that separates every three digits and a period that separates the decimal fraction.
COMMA*w*.*d*	writes numeric values with a period that separates every three digits and a comma that separates the decimal fraction.
COMMAX*w*.*d*	writes numeric values with a leading dollar sign, a comma that separates every three digits, and a period that separates the decimal fraction.
DOLLAR*w*.*d*	writes numeric values with a leading euro symbol (€), a period that separates every three digits, and a comma that separates the decimal fraction.
EUROX*w*.*d*	

SAS Formats

Selected SAS formats:

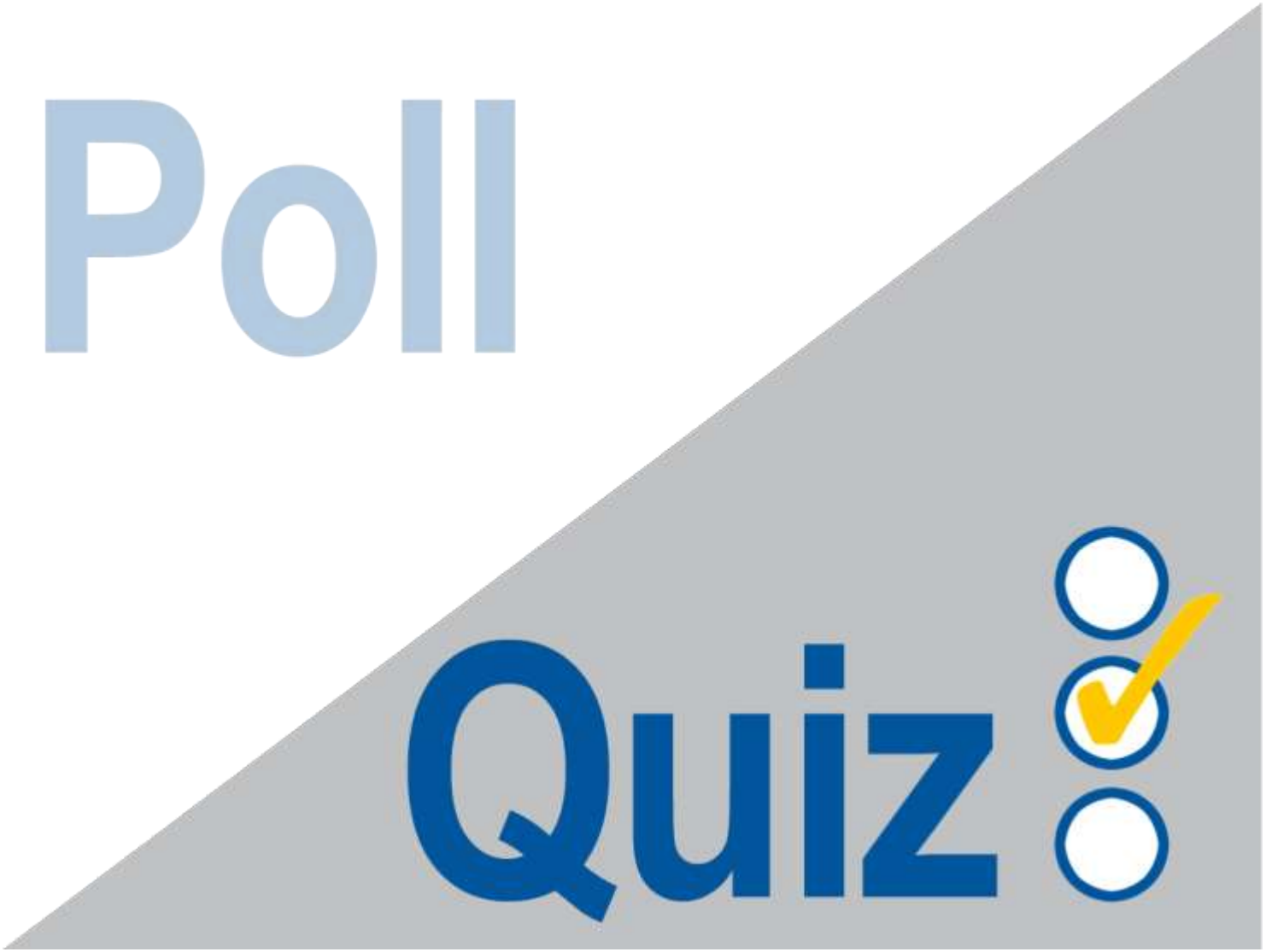
Format	Stored Value	Displayed Value
$4.	Programming	Prog
12.	27134.2864	27134
12.2	27134.2864	27134.29
COMMA12.2	27134.2864	27,134.29
COMMAX12.2	27134.2864	27.134,29
DOLLAR12.2	27134.2864	$27,134.29
EUROX12.2	27134.2864	€27.134,29

SAS Formats

If you do not specify a format width that is large enough to accommodate a numeric value, the displayed value is automatically adjusted to fit into the width.

Format	Stored Value	Displayed Value
DOLLAR12.2	27134.2864	$27,134.29
DOLLAR9.2	27134.2864	$27134.29
DOLLAR8.2	27134.2864	27134.29
DOLLAR5.2	27134.2864	27134
DOLLAR4.2	27134.2864	27E3

Poll



5.06 Quiz

Which numeric format writes standard numeric data with leading zeros?

Documentation on formats can be found in the SAS Help and Documentation from the Contents tab (**SAS Products** ⇨ **Base SAS** ⇨ **SAS 9.4 Formats and Informats: Reference** ⇨ **SAS Formats** ⇨ **Dictionary of Formats** ⇨ **Formats by Category**).

5.06 Quiz – Correct Answer

Which numeric format writes standard numeric data with leading zeros?

Zw.d

The *Zw.d* format is similar to the *w.d* format except that *Zw.d* pads right-aligned output with zeros instead of blanks.


Question
&
Answer

SAS Date Formats

SAS date formats display SAS date values in standard date forms.

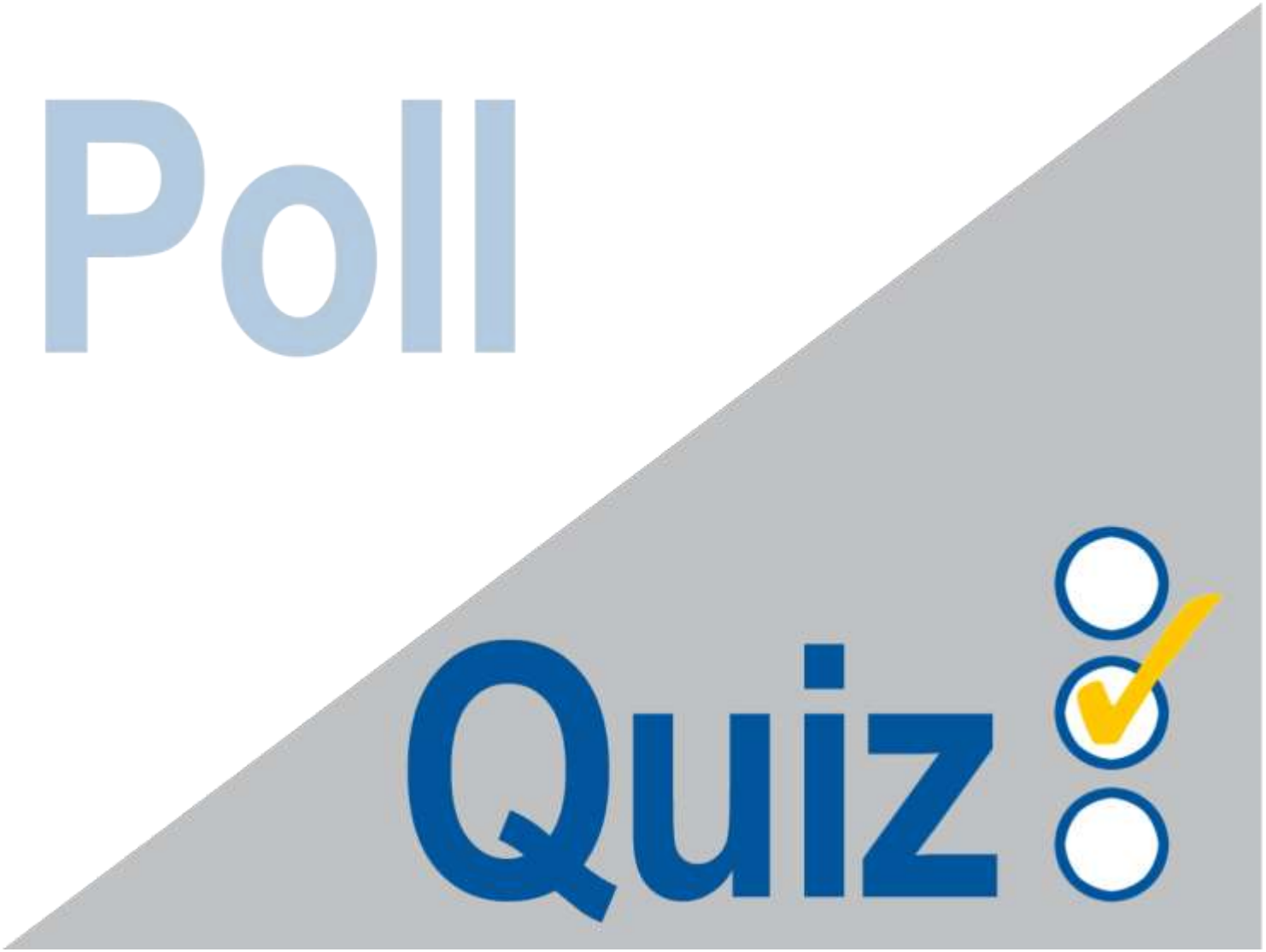
Format	Stored Value	Displayed Value
MMDDYY6.	0	010160
MMDDYY8.	0	01/01/60
MMDDYY10.	0	01/01/1960
DDMMYY6.	365	311260
DDMMYY8.	365	31/12/60
DDMMYY10.	365	31/12/1960

SAS Date Formats

Additional date formats:

Format	Stored Value	Displayed Value
DATE7.	-1	31DEC59
DATE9.	-1	31DEC1959
WORDDATE.	0	January 1, 1960
WEEKDATE.	0	Friday, January 1, 1960
MONYY7.	0	JAN1960
YEAR4.	0	1960

Poll



5.07 Quiz

Which FORMAT statement creates the output?

a.
```
format Birth_Date Hire_Date ddmmyy9.
       Term_Date mmyy7.;
```

b.
```
format Birth_Date Hire_Date ddmmyyyy.
       Term_Date mmmyyyy.;
```

c.
```
format Birth_Date Hire_Date ddmmyy10.
       Term_Date monyy7.;
```

Output

Birth_Date	Hire_Date	Term_Date
21/05/1969	15/10/1992	MAR2007

5.07 Quiz – Correct Answer

Which FORMAT statement creates the output?

a.
```
format Birth_Date Hire_Date ddmmyy9.
       Term_Date mmyy7.;
```

b.
```
format Birth_Date Hire_Date ddmmyyyy.
       Term_Date mmmyyyy.;
```

c.
```
format Birth_Date Hire_Date ddmmyy10.
       Term_Date monyy7.;
```

Output

Birth_Date	Hire_Date	Term_Date
21/05/1969	15/10/1992	MAR2007

Business Scenario Part 3

Include formats in the descriptor portion of **Work.subset1**.

```
data work.subset1;
   set orion.sales;
   where Country='AU' and
         Job_Title contains 'Rep';
   keep First_Name Last_Name Salary
        Job_Title Hire_Date;
   label Job_Title='Sales Title'
         Hire_Date='Date Hired';
   format Salary commax8. Hire_Date ddmmyy10.;
run;
```

Business Scenario Part 3

```
proc contents data=work.subset1;
run;
```

Partial PROC CONTENTS Output

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Label
1	First_Name	Char	12		
5	Hire_Date	Num	8	DDMMYY10.	Date Hired
4	Job_Title	Char	25		Sales Title
2	Last_Name	Char	18		
3	Salary	Num	8	COMMAX8.	

Business Scenario Part 3

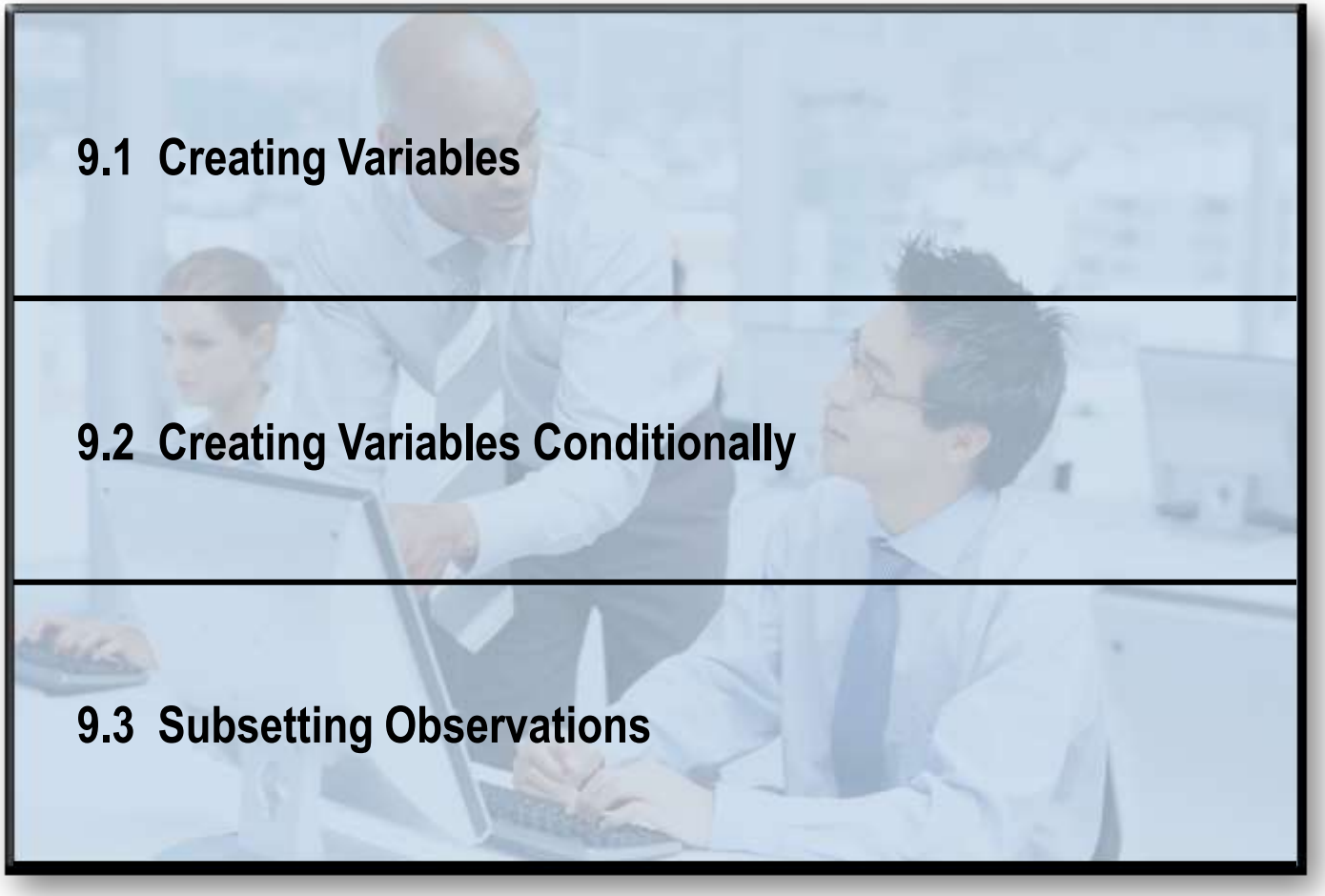
```
proc print data=work.subset1 label;
run;
```

Partial PROC PRINT Output

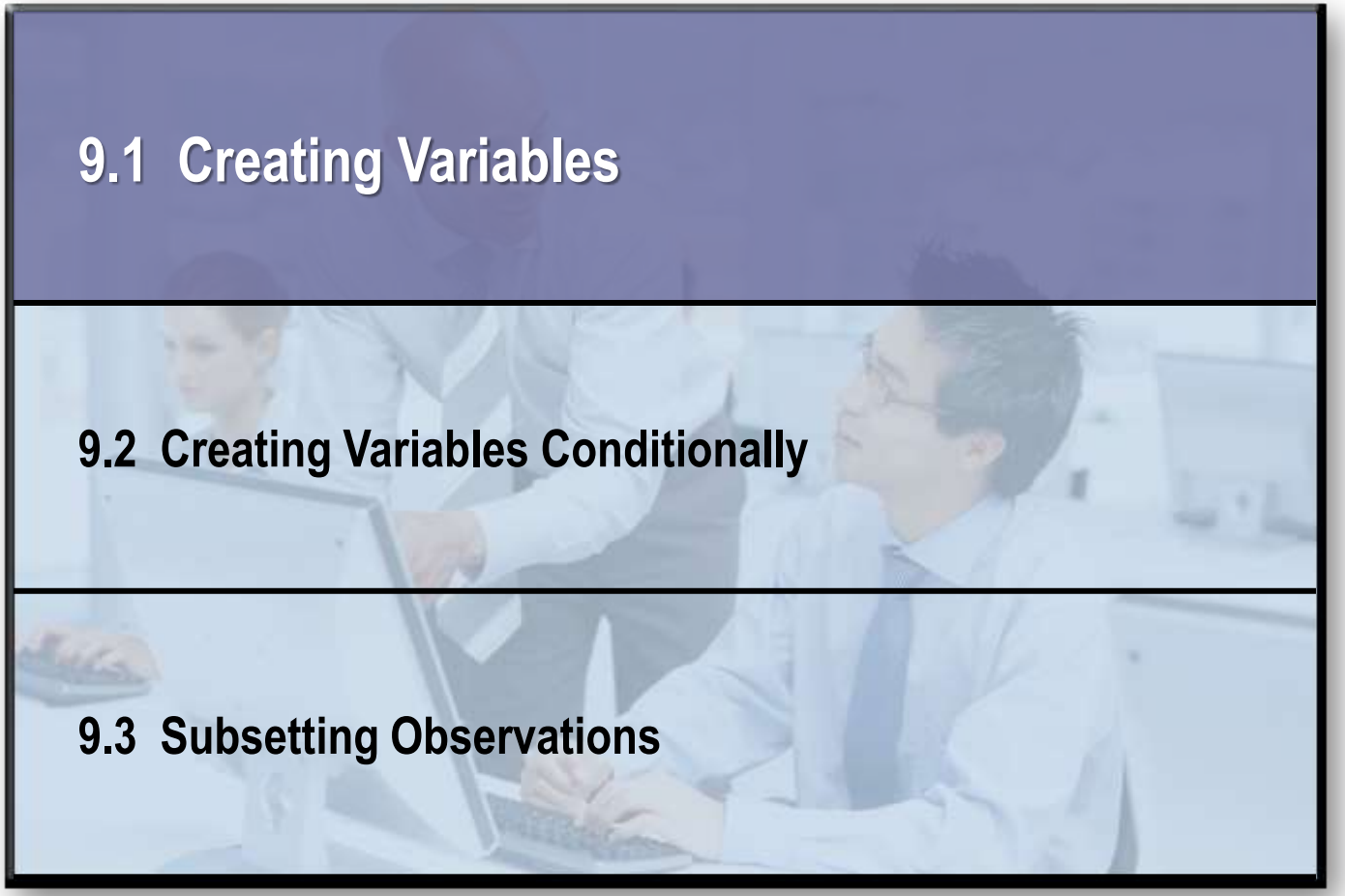
Obs	First_ Name	Last_Name	Salary	Sales Title	Date Hired
1	Irenie	Elvish	26.600	Sales Rep. II	01/01/1974
2	Christina	Ngan	27.475	Sales Rep. II	01/07/1978
3	Kimiko	Hotstone	26.190	Sales Rep. I	01/10/1985
4	Lucian	Daymond	26.480	Sales Rep. I	01/03/1979
5	Fong	Hofmeister	32.040	Sales Rep. IV	01/03/1979
6	Satyakam	Denny	26.780	Sales Rep. II	01/08/2006
7	Sharryn	Clarkson	28.100	Sales Rep. II	01/11/1998
8	Monica	Kletschkus	30.890	Sales Rep. IV	01/11/2006
9	Alvin	Roebuck	30.070	Sales Rep. III	01/10/1985
10	Kevin	Lyon	26.955	Sales Rep. I	01/05/2006
11	Marinus	Surawski	26.910	Sales Rep. I	01/01/2003
12	Fancine	Kaiser	28.525	Sales Rep. III	01/10/1978


Question
&
Answer

Chapter 9: Manipulating Data



Chapter 9: Manipulating Data



Objectives

- Create SAS variables with the assignment statement in the DATA step.
- Create data values by using operators including SAS functions.
- Subset variables by using the DROP and KEEP statements.
- Examine the compilation and execution phases of the DATA step when you read a SAS data set.

Business Scenario

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

Work.comp must include the following new variables:

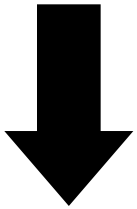
- **Bonus**, which is equal to a constant 500
- **Compensation**, which is the combination of the employee's salary and bonus
- **BonusMonth**, which is equal to the month that the employee was hired

Work.comp must not include the **Gender**, **Salary**, **Job_Title**, **Country**, **Birth_Date**, and **Hire_Date** variables from **orion.sales**.

Business Scenario

Partial **orion.sales**

Employee_ ID	First_ Name	Last_ Name	Gender	Salary	Job_ Title	Country	Birth_ Date	Hire_ Date
120102	Tom	Zhou	M	108255	Sales Manager	AU	3510	10744
120103	Wilson	Dawes	M	87975	Sales Manager	AU	-3996	5114
120121	Irenie	Elvish	F	26600	Sales Rep. II	AU	-5630	5114



Partial **Work.comp**

Employee_ ID	First_ Name	Last_ Name	Bonus	Compensation	Bonus Month
120102	Tom	Zhou	500	108755	6
120103	Wilson	Dawes	500	88475	1
120121	Irenie	Elvish	500	27100	1

Assignment Statements

Assignment statements are used in the DATA step to update existing variables or create new variables.

```
DATA output-SAS-data-set;
      SET input-SAS-data-set;
      variable = expression;
RUN;
```

```
DATA output-SAS-data-set;
      INFILE 'raw-data-file-name';
      INPUT specifications;
      variable = expression;
RUN;
```

Assignment Statements

The *assignment statement* evaluates an expression and assigns the resulting value to a variable.

General form of the assignment statement:

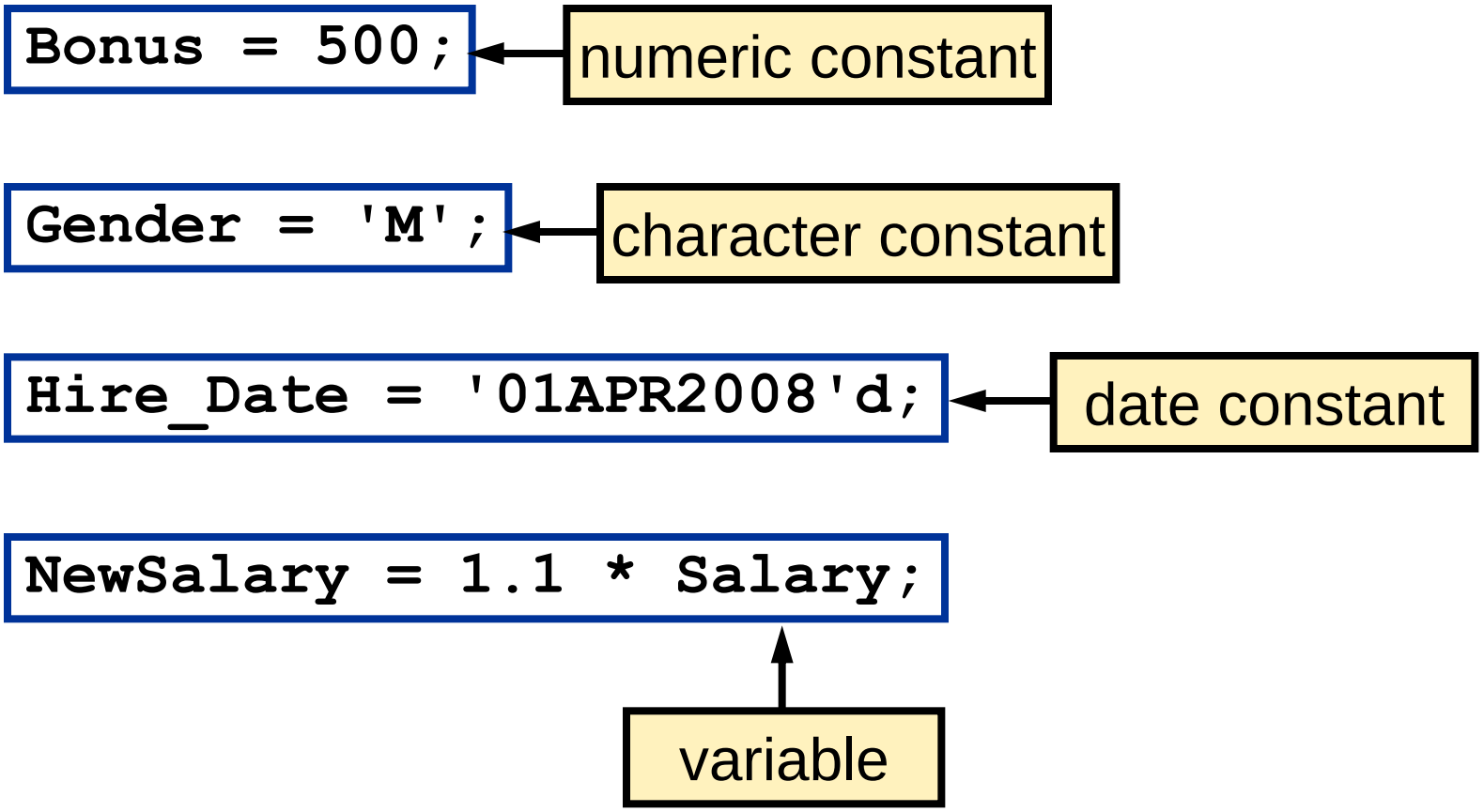
variable = *expression*;

- *variable* names an existing or new variable.
- *expression* is a sequence of operands and operators that form a set of instructions that produce a value.

Operands

Operands are constants (character, numeric, or date) and variables (character or numeric).

Examples:



Two Digit Years

What happens if you enter this:

```
Payoff_Date = '01APR20'd;
```

date constant

Obs	Payoff_Date
1	-14519

YEARCUTOFF

- System option that specifies first year of 100 year span for interpreting two-digit years
- Default in 9.4 = 1926
- Use four-digit years to avoid misinterpretation

Operators

Operators are symbols that represent an arithmetic calculation and SAS functions.

Examples:

```
Revenue = Quantity * Price;
```

```
NewCountry = upcase(Country);
```

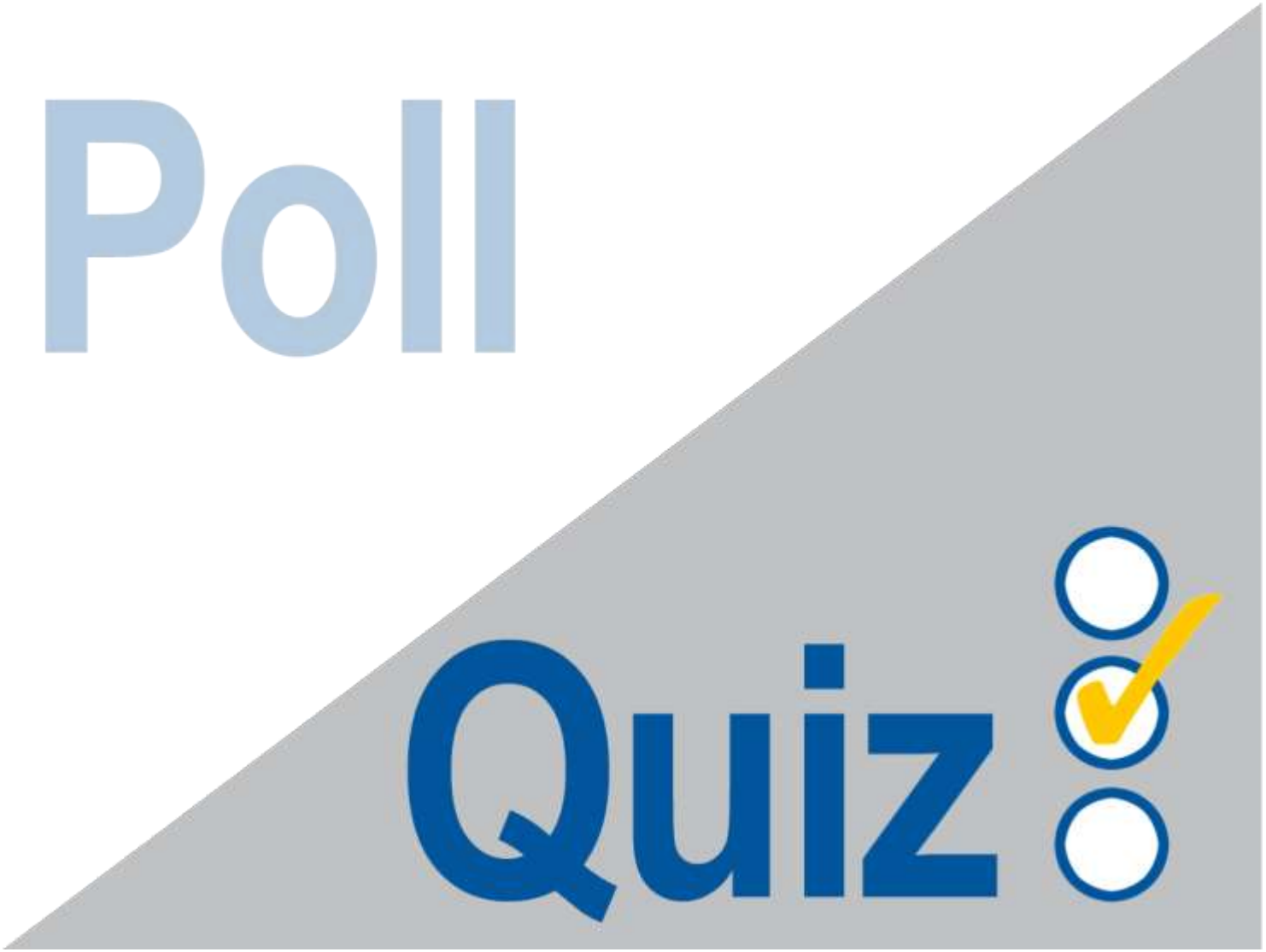
Arithmetic Operators

Arithmetic operators indicate that an arithmetic calculation is performed.

Symbol	Definition	Priority
**	exponentiation	I
-	negative prefix	I
*	multiplication	II
/	division	II
+	addition	III
-	subtraction	III

- If a missing value is an operand for an arithmetic operator, the result is a missing value.

Poll



9.01 Quiz

What is the result of the assignment statement?

a. . (missing)
b. 0
c. 7
d. 9

```
num = 4 + 10 / 2;
```

9.01 Quiz – Correct Answer

What is the result of the assignment statement?

a. . (missing)
b. 0
c. 7
d. 9

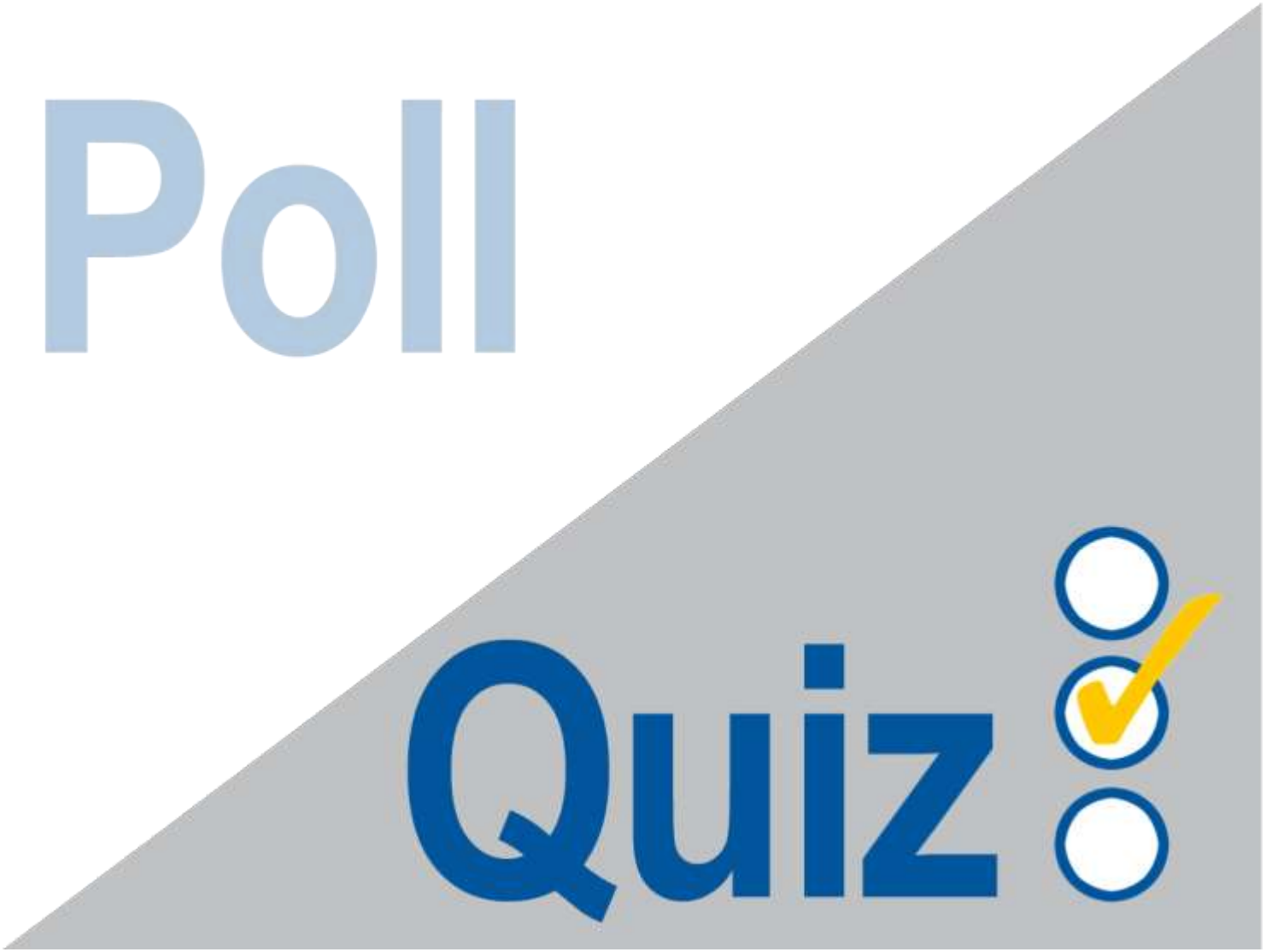
```
num = 4 + 10 / 2;
```

The order of operations from left to right is division and multiplication followed by addition and subtraction.

Parentheses can be used to control the order of operations.

```
num = (4 + 10) / 2;
```

Poll



9.02 Quiz

What is the result of the assignment statement given the values of **`var1`** and **`var2`**?

a. . (missing)
b. 0
c. 5
d. 10

```
num = var1 + var2 / 2;
```

var1	var2
.	10

9.02 Quiz – Correct Answer

What is the result of the assignment statement given the values of **`var1`** and **`var2`**?

a. . (missing)
b. 0
c. 5
d. 10

```
num = var1 + var2 / 2;
```

var1	var2
.	10

If an operand is missing for an arithmetic operator, the result is missing.

SAS Functions

A SAS *function* is a routine that returns a value that is determined from specified arguments.

Some SAS functions manipulate character values, compute descriptive statistics, or manipulate SAS date values.

General form of a SAS function:

function-name(*argument1*, *argument2*, …)

- Depending on the function, zero, one, or many arguments are used.
- Arguments are separated with commas.

Descriptive Statistics Function

The *SUM function* returns the sum of the arguments.

General form of the SUM function:

SUM(*argument1*,*argument2*, ...**)**

- The arguments must be numeric values.
- Missing values are ignored by some of the descriptive statistics functions.

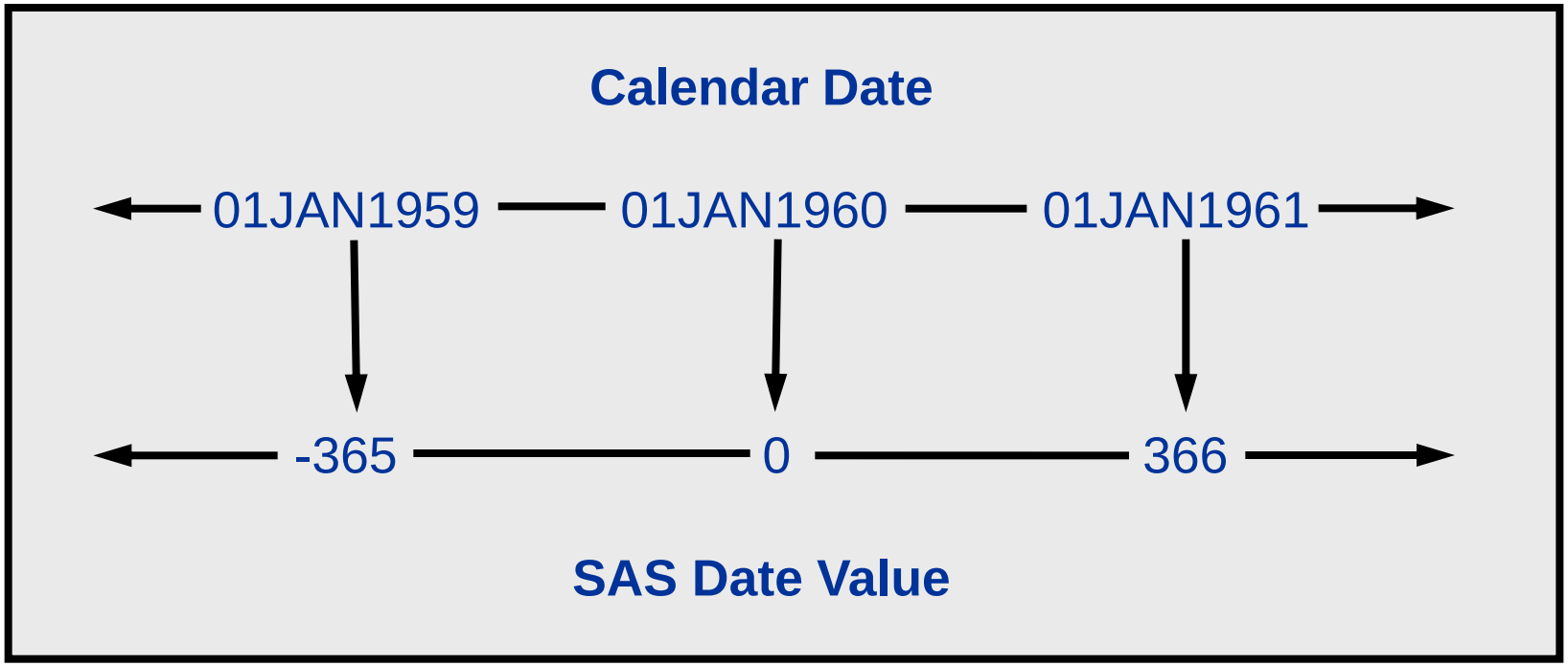
Example:

```
Compensation=sum(Salary,Bonus);
```

Date Functions

SAS date functions can be used to

- extract information from SAS date values
- create SAS date values.



Date Functions – Extracting Information

YEAR(*SAS-date*)	extracts the year from a SAS date and returns a four-digit value for year.
QTR(*SAS-date*)	extracts the quarter from a SAS date and returns a number from 1 to 4.
MONTH(*SAS-date*)	extracts the month from a SAS date and returns a number from 1 to 12.
DAY(*SAS-date*)	extracts the day of the month from a SAS date and returns a number from 1 to 31.
WEEKDAY(*SAS-date*)	extracts the day of the week from a SAS date and returns a number from 1 to 7, where 1 represents Sunday, and so on.

Example:

```
BonusMonth=month(Hire_Date);
```

Date Functions – Creating SAS Dates

TODAY()	returns the current date as a SAS date value.
MDY(*month*,*day*,*year*)	returns a SAS date value from numeric month, day, and year values.

Example:

```
AnnivBonus=mdy(month(Hire_Date),15,2008);
```


Question
&
Answer

Business Scenario

Create **Bonus**, **Compensation**, and **BonusMonth**.

```
data work.comp;
    set orion.sales;
    Bonus=500;
    Compensation=sum(Salary,Bonus);
    BonusMonth=month(Hire_Date);
run;
```

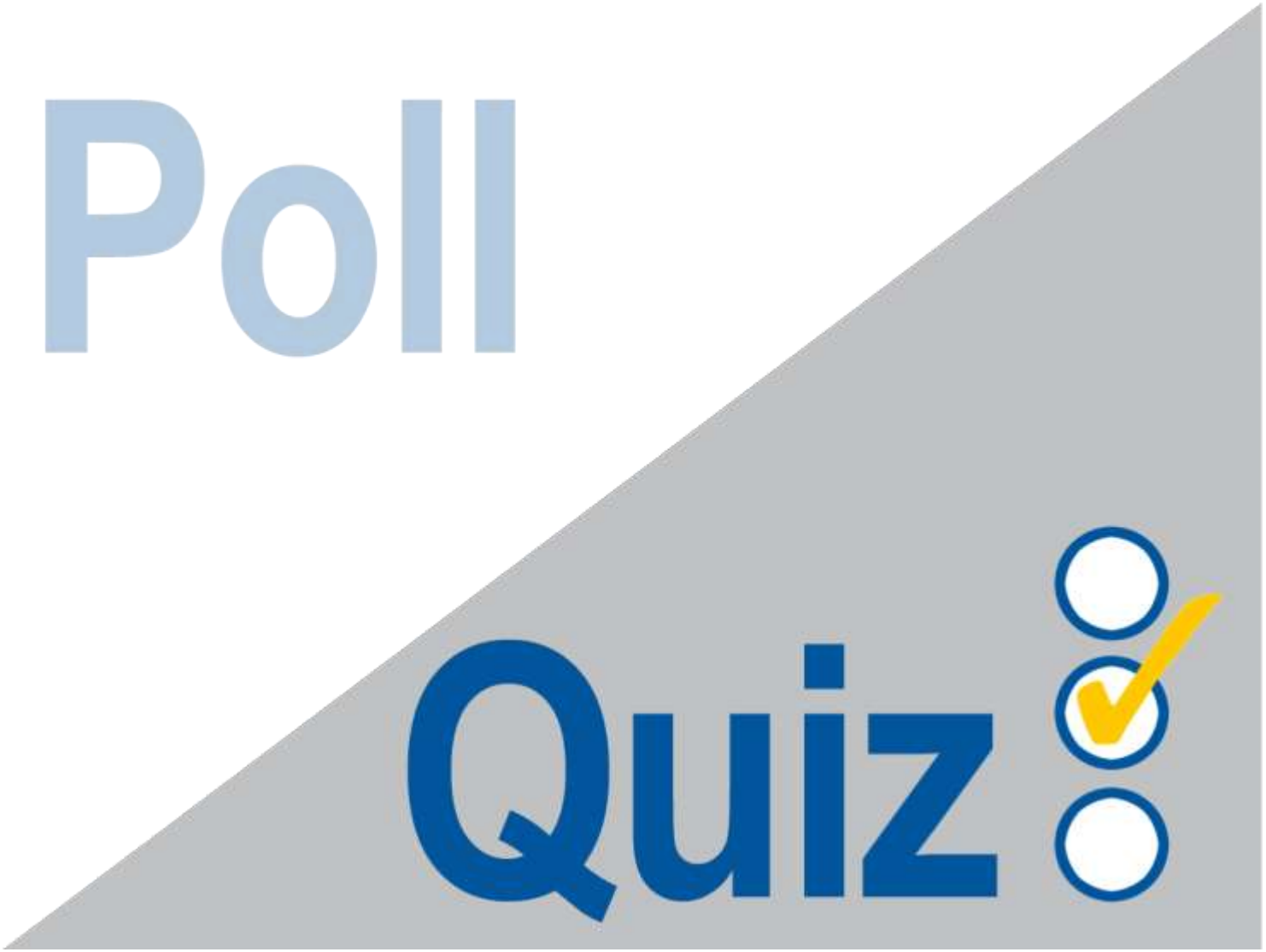
```
1700  data work.comp;
1701     set orion.sales;
1702     Bonus=500;
1703     Compensation=sum(Salary,Bonus);
1704     BonusMonth=month(Hire_Date);
1705  run;

NOTE: There were 165 observations read from the data set ORION.SALES.
NOTE: The data set WORK.COMP has 165 observations and 12 variables.
```

orion.sales
has 9 variables.

p109d01

Poll



9.03 Quiz

What statement needs to be added to the DATA step to eliminate six of the 12 variables?

9.03 Quiz – Correct Answer

What statement needs to be added to the DATA step to eliminate six of the 12 variables?

the DROP or KEEP statement

Business Scenario

Drop **Gender**, **Salary**, **Job_Title**, **Country**, **Birth_Date**, and **Hire_Date**.

```
data work.comp;
   set orion.sales;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
   drop Gender Salary Job_Title
        Country Birth_Date Hire_Date;
run;
```

Partial SAS Log

```
NOTE: There were 165 observations read from the data set ORION.SALES.
NOTE: The data set WORK.COMP has 165 observations and 6 variables.
```

p109d01

Business Scenario

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

Partial PROC PRINT Output

Obs	Employee_ID	First_ Name	Last_Name	Bonus	Compensation	Bonus Month
1	120102	Tom	Zhou	500	108755	6
2	120103	Wilson	Dawes	500	88475	1
3	120121	Irenie	Elvish	500	27100	1
4	120122	Christina	Ngan	500	27975	7
5	120123	Kimiko	Hotstone	500	26690	10
6	120124	Lucian	Daymond	500	26980	3
7	120125	Fong	Hofmeister	500	32540	3
8	120126	Satyakam	Denny	500	27280	8
9	120127	Sharryn	Clarkson	500	28600	11
10	120128	Monica	Kletschkus	500	31390	11

p109d01


Poll
Quiz

Setup for the Poll

- Submit program **p109a01**.
- Verify the results.

```
data work.comp;
    set orion.sales;
    drop Gender Salary Job_Title
         Country Birth_Date Hire_Date;
    Bonus=500;
    Compensation=sum(Salary,Bonus);
    BonusMonth=month(Hire_Date);
run;
```

9.04 Poll

Are the correct results produced when the DROP statement is placed after the SET statement?

- Yes
- No

9.04 Poll – Correct Answer

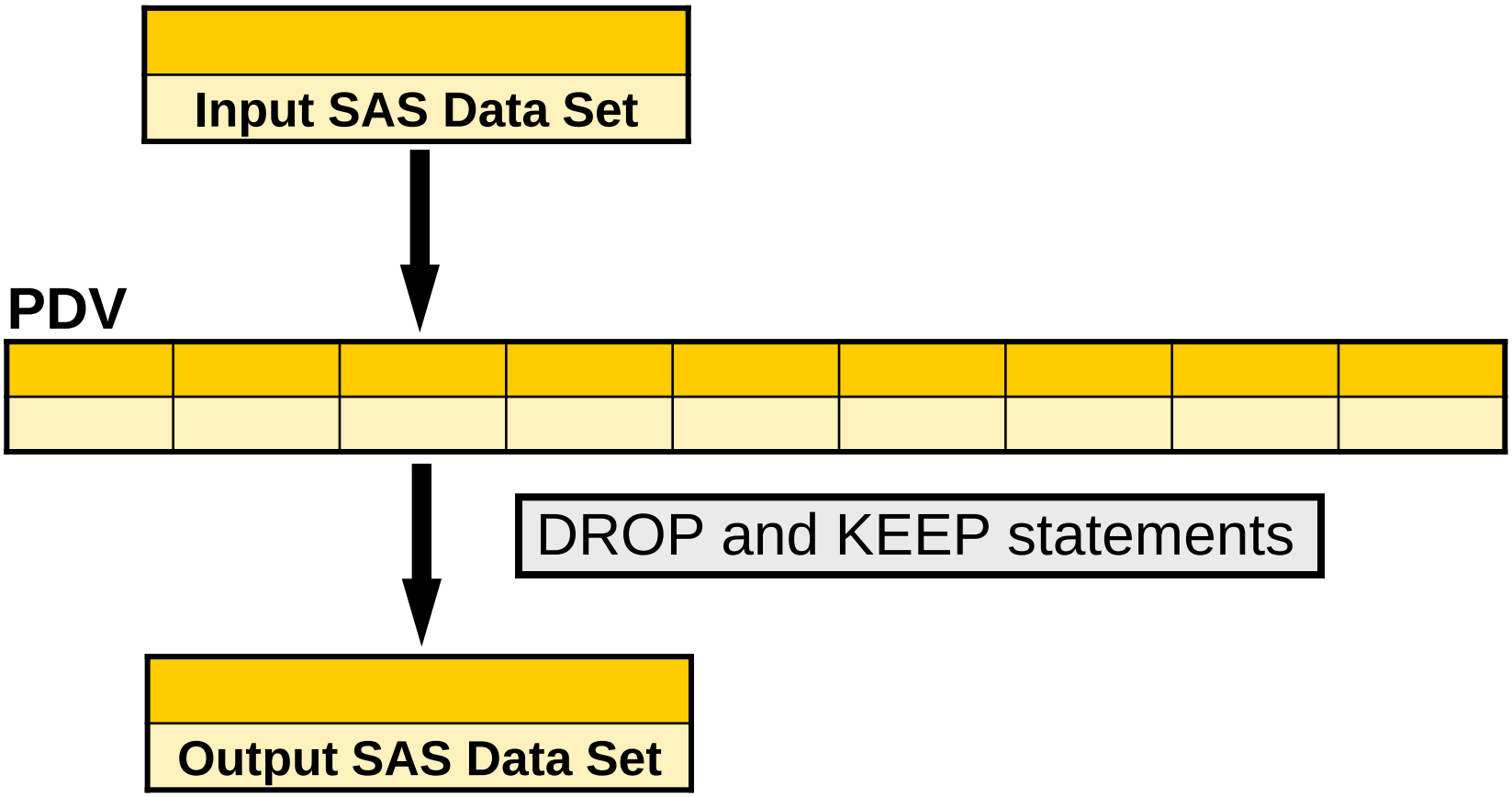
Are the correct results produced when the DROP statement is placed after the SET statement?

- ❍ Yes
- ❍ No

Yes, the DROP statement specifies the names of the variables to omit from the output data set.

Processing the DROP and KEEP Statements

The DROP and KEEP statements select variables **after** they are brought into the program data vector.



Compilation

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```


...

Compilation

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	First_Name	Last_Name	Gender	Salary	Job_Title
N 8	$ 12	$ 18	$ 1	N 8	$ 25

Country	Birth_Date	Hire_Date
$ 2	N 8	N 8

...

Compilation

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	First_Name	Last_Name	Gender	Salary	Job_Title
N 8	$ 12	$ 18	$ 1	N 8	$ 25

Country	Birth_Date	Hire_Date	Bonus
$ 2	N 8	N 8	N 8

...

Compilation

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	First_Name	Last_Name	Gender	Salary	Job_Title
N 8	$ 12	$ 18	$ 1	N 8	$ 25

Country	Birth_Date	Hire_Date	Bonus	Compensation
$ 2	N 8	N 8	N 8	N 8

...

Compilation

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	First_Name	Last_Name	Gender	Salary	Job_Title
N 8	$ 12	$ 18	$ 1	N 8	$ 25

Country	Birth_Date	Hire_Date	Bonus	Compensation	BonusMonth
$ 2	N 8	N 8	N 8	N 8	N 8

...

Compilation

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	First_Name	Last_Name	Gender	Salary	Job_Title
N 8	$ 12	$ 18	D $ 1	D N 8	D $ 25

Country	Birth_Date	Hire_Date	Bonus	Compensation	BonusMonth
D $ 2	D N 8	D N 8	N 8	N 8	N 8

...

Compilation

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	First_Name	Last_Name	Gender	Salary	Job_Title
N 8	$ 12	$ 18	D $ 1	D N 8	D $ 25

Country	Birth_Date	Hire_Date	Bonus	Compensation	BonusMonth
D $ 2	D N 8	D N 8	N 8	N 8	N 8

Descriptor Portion Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth
N 8	$ 12	$ 18	N 8	N 8	N 8

Execution

Partial orion.sales

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
    set ori
    drop Ge                    itle
         Co
         Hire_Date;
    Bonus=500;
    Compensation=sum(Salary,Bonus);
    BonusMonth=month(Hire_Date);
run;
```

Initialize PDV

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
.				.	.	.	.

Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth

...

Execution

Partial orion.sales

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date
        Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120102		M		10744	.	.	.

Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth

...

Execution

Partial orion.sales

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date
        Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120102		M		10744	500	.	.

Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth

...

Execution

Partial orion.sales

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date
        Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120102		M		10744	500	108755	.

Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth

...

Execution

Partial orion.sales

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date
        Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120102		M		10744	500	108755	6

Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth

...

Execution

Partial `orion.sales`

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
    set orion.sales;
    drop Gender Salary Job_Title
         Country Birth_Date
         Hire_Date;
    Bonus=500;
    Compensation=sum(Salary,Bonus);
    BonusMon                        ;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120102		M		10744	500	108755	6

`Work.comp`

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth
120102	Tom	Zhou	500	108755	6

...

Execution

Partial orion.sales

Employee_ID	...	Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
    set ori
    drop Ge                    itle
         Co
         Hire_Date;
    Bonus=500;
    Compensation=sum(Salary,Bonus);
    BonusMonth=month(Hire_Date);
run;
```

Reinitialize PDV

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120102		M		10744	.	.	.

Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth
120102	Tom	Zhou	500	108755	6

...

Execution

Partial orion.sales

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
    set orion.sales;
    drop Gender Salary Job_Title
         Country Birth_Date
         Hire_Date;
    Bonus=500;
    Compensation=sum(Salary,Bonus);
    BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120103		M		5114	.	.	.

Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth
120102	Tom	Zhou	500	108755	6

...

Execution

Partial orion.sales

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date
        Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120103		M		5114	500	.	.

Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth
120102	Tom	Zhou	500	108755	6

...

Execution

Partial orion.sales

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date
        Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120103		M		5114	500	88475	.

Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth
120102	Tom	Zhou	500	108755	6

...

Execution

Partial orion.sales

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date
        Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120103		M		5114	500	88475	1

Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth
120102	Tom	Zhou	500	108755	6

...

Execution

Partial orion.sales

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

```
data work.comp;
   set orion.sales;
   drop Gender Salary Job_Title
        Country Birth_Date
        Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMon                        ;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120103		M		5114	500	88475	1

Work.comp

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth
120102	Tom	Zhou	500	108755	6
120103	Wilson	Dawes	500	88475	1

Execution

Partial `orion.sales`

Employee_ID		Hire_Date
120102		10744
120103	...	5114
120121		5114
120122		6756

Continue until EOF

```
data work.comp;
                    Job_Title
        Country Birth_Date
        Hire_Date;
   Bonus=500;
   Compensation=sum(Salary,Bonus);
   BonusMonth=month(Hire_Date);
run;
```

PDV

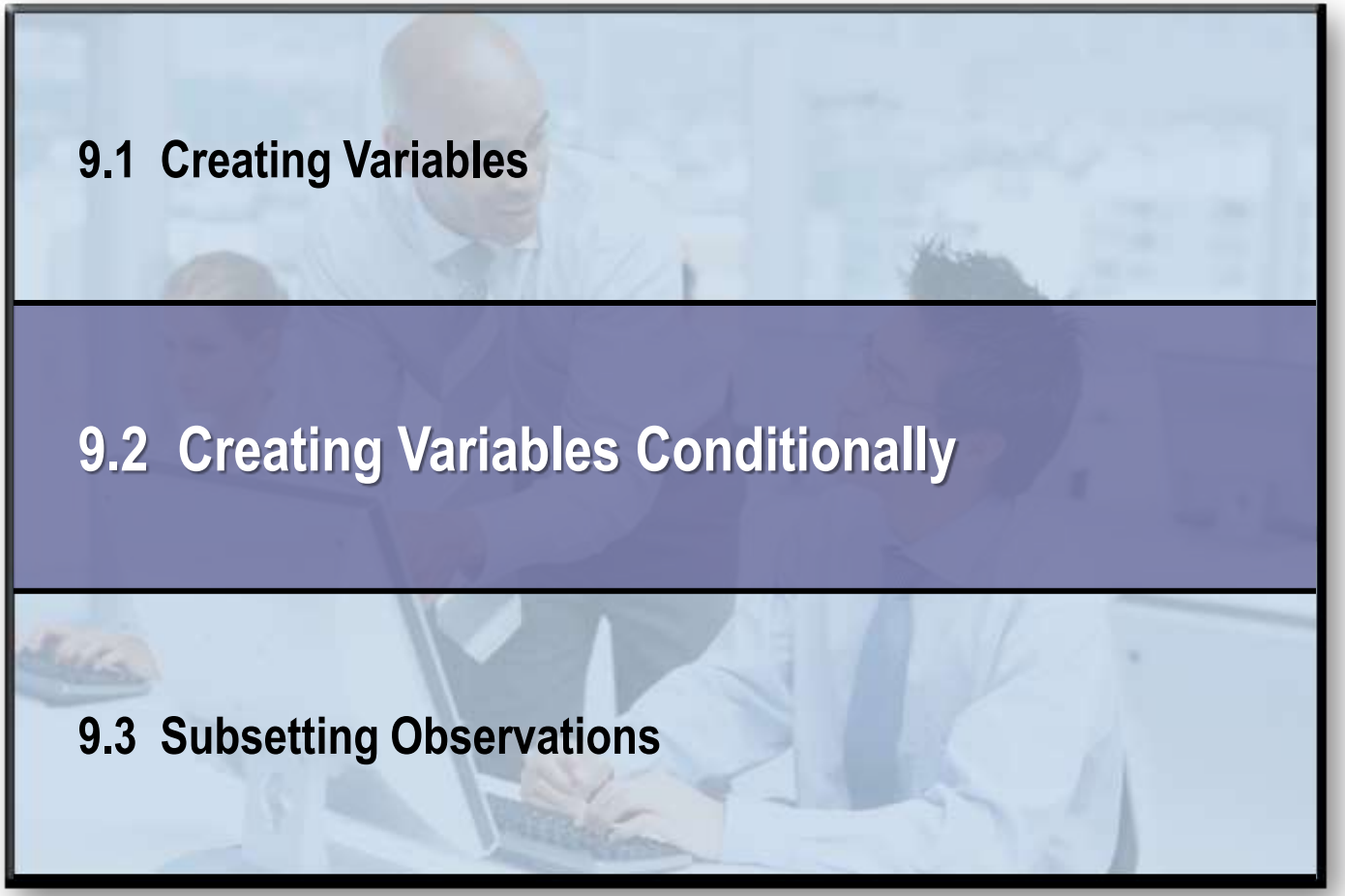
Employee_ID	...	Gender	...	Hire_Date	Bonus	Compensation	BonusMonth
120103		M		5114	500	88475	1

`Work.comp`

Employee_ID	First_Name	Last_Name	Bonus	Compensation	BonusMonth
120102	Tom	Zhou	500	108755	6
120103	Wilson	Dawes	500	88475	1


Question
&
Answer

Chapter 9: Manipulating Data



Objectives

- Execute statements conditionally by using IF-THEN and IF-THEN DO statements.
- Give alternate actions if the previous THEN clause is not executed by using the ELSE statement.
- Control the length of character variables by using the LENGTH statement.

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.

IF-THEN Statements

The *IF-THEN statement* executes a SAS statement for observations that meet specific conditions.

General form of the IF-THEN statement:

IF *expression* **THEN** *statement*;

- *expression* is a sequence of operands and operators that form a set of instructions that define a condition for selecting observations.
- *statement* is any executable statement such as the assignment statement.

IF-THEN/ELSE Statements

The optional *ELSE statement* gives an alternate action if the previous THEN clause is not executed.

General form of the IF-THEN/ELSE statements:

IF *expression* **THEN** *statement*;
ELSE IF *expression* **THEN** *statement*;

- Using IF-THEN statements **without** the ELSE statement causes SAS to evaluate all IF-THEN statements.
- Using IF-THEN statements **with** the ELSE statement causes SAS to execute IF-THEN statements until it encounters the first true statement.

Business Scenario

Create the new variable **Bonus**.

```
data work.bonus;
    set orion.sales;
    if Country='US' then Bonus=500;
    else if Country='AU' then Bonus=300;
run;
```

```
1819  data work.bonus;
1820     set orion.sales;
1821     if Country='US' then Bonus=500;
1822     else if Country='AU' then Bonus=300;
1823  run;

NOTE: There were 165 observations read from the data set ORION.SALES.
NOTE: The data set WORK.BONUS has 165 observations and 10 variables
```


p109d03

Business Scenario

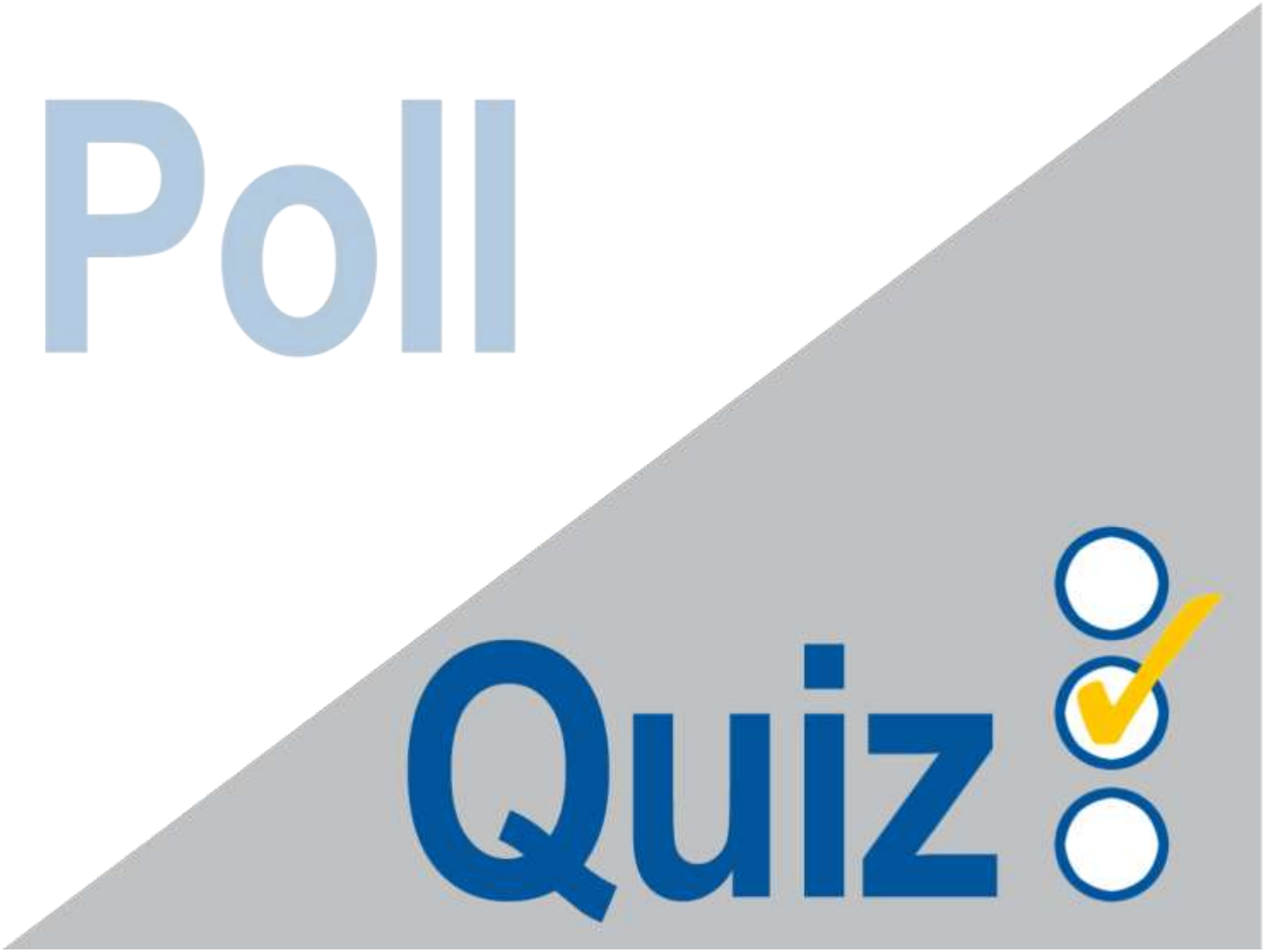
```
proc print data=work.bonus;
   var First_Name Last_Name Country Bonus;
run;
```

Partial PROC PRINT Output

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

p109d03

Poll



9.05 Quiz

- Submit program **p109a02**.
- Review the results.
- Why are some of the **Bonus** values missing in the PROC PRINT output for **orion.nonsales**?

9.05 Quiz – Correct Answer

Why are some of the **Bonus** values missing in the PROC PRINT output for **orion.nonsales**?

Country has mixed case values in the orion.nonsales data set.

The UPCASE function will correct the issue.

```
data work.bonus;
   set orion.nonsales;
   if upcase(Country)='US'
      then Bonus=500;
   else if upcase(Country)='AU'
         then Bonus=300;
run;
```

p109a02s

ELSE Statements

The conditional clause does not have to be in an ELSE statement.

For example:

```
data work.bonus;
   set orion.sales;
   if Country='US' then Bonus=500;
   else Bonus=300;
run;
```

All observations not equal to US get a bonus of 300.