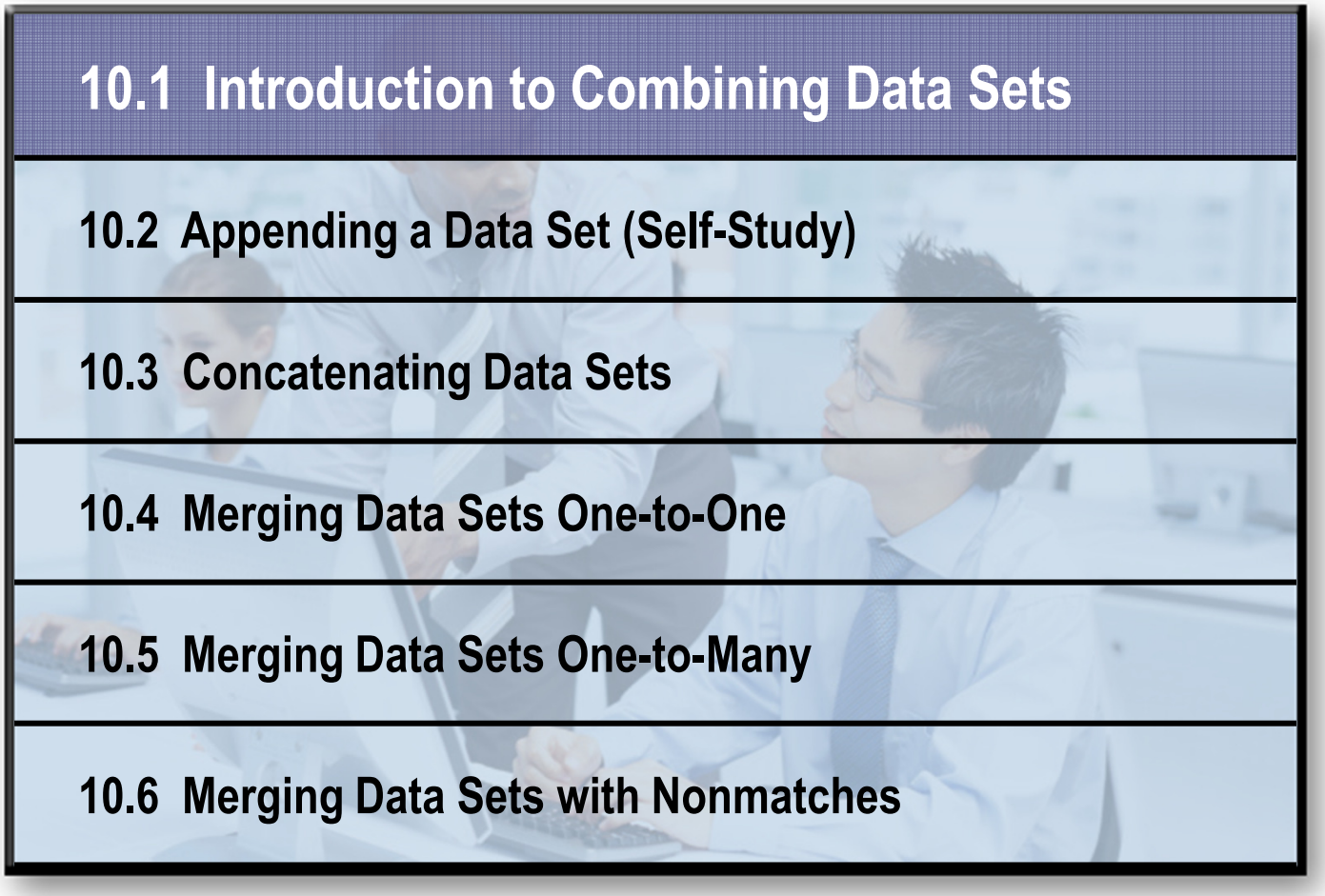
Chapter 10: Combining SAS Data Sets



10.1 Introduction to Combining Data Sets

10.2 Appending a Data Set (Self-Study)

10.3 Concatenating Data Sets

10.4 Merging Data Sets One-to-One

10.5 Merging Data Sets One-to-Many

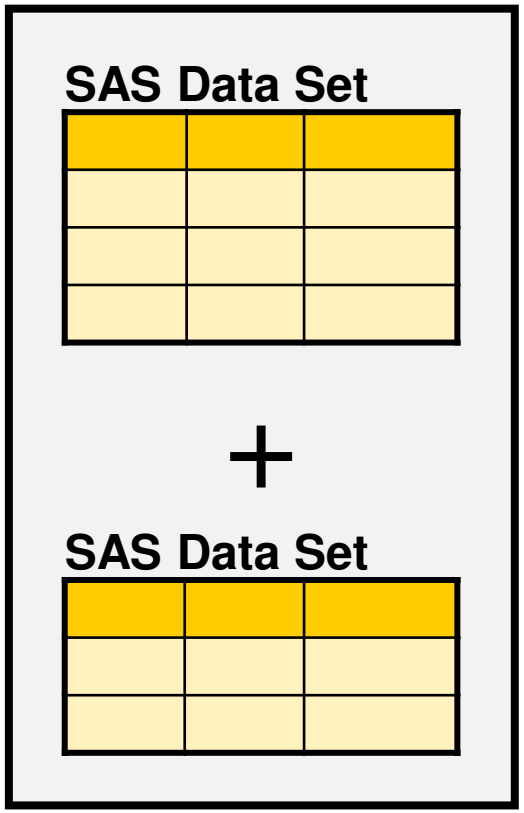
10.6 Merging Data Sets with Nonmatches

Objectives

- Define the methods for combining SAS data sets.

Appending and Concatenating

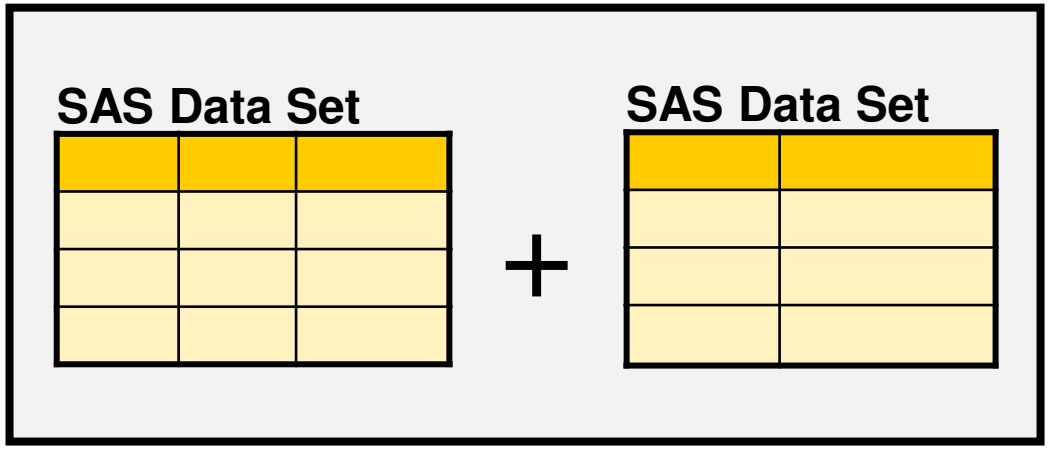
Appending and concatenating involves combining SAS data sets, one after the other, into a single SAS data set.



- *Appending* adds the observations in the second data set directly to the end of the original data set.
- *Concatenating* copies all observations from the first data set and then copies all observations from one or more successive data sets into a new data set.

Merging

Merging involves combining observations from two or more SAS data sets into a single observation in a new SAS data set.



Observations can be merged based on their positions in the original data sets or merged by one or more common variables.

Example: Appending a Data Set

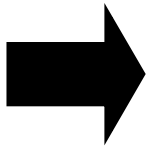
One data set is appended to a master data set.

Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007

Emps2008

First	Gender	HireYear
Brett	M	2008
Renee	F	2008



Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007
Brett	M	2008
Renee	F	2008

Example: Concatenating Data Sets

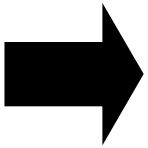
Two data sets are concatenated to create a new data set.

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France



EmpsAll

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark
Pierre	M	France
Sophie	F	France

Example: Merging Data Sets

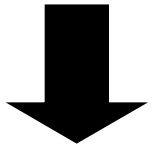
Two data sets are merged to create a new data set.

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneH

EmpID	Phone
121150	+61(2)5555-1793
121151	+61(2)5555-1849
121152	+61(2)5555-1665



EmpsAUH

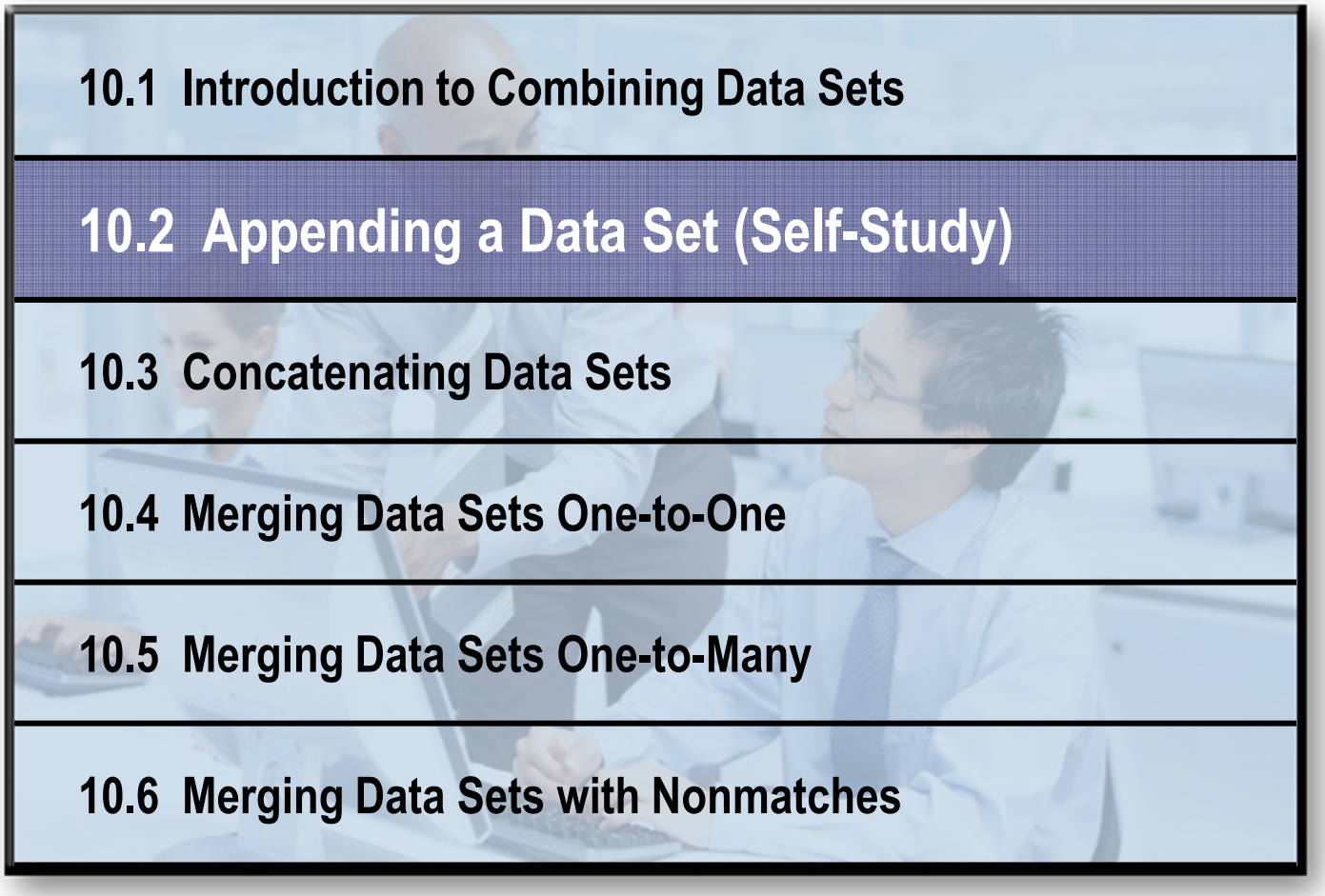
First	Gender	EmpID	Phone
Togar	M	121150	+61(2)5555-1793
Kylie	F	121151	+61(2)5555-1849
Birin	M	121152	+61(2)5555-1665

10.01 Quiz

Which method (appending, concatenating, or merging) should be used for the given business scenario?

	Business Scenario	Method
1	The **JanSales**, **FebSales**, and **MarSales** data sets need to be combined to create the **Qtr1Sales** data set.	
2	The **Sales** data set needs to be combined with the **Target** data set by **month** to compare the sales data to the target data.	
3	The **OctSales** data set needs to be added to the **YTD** data set.	

Chapter 10: Combining SAS Data Sets

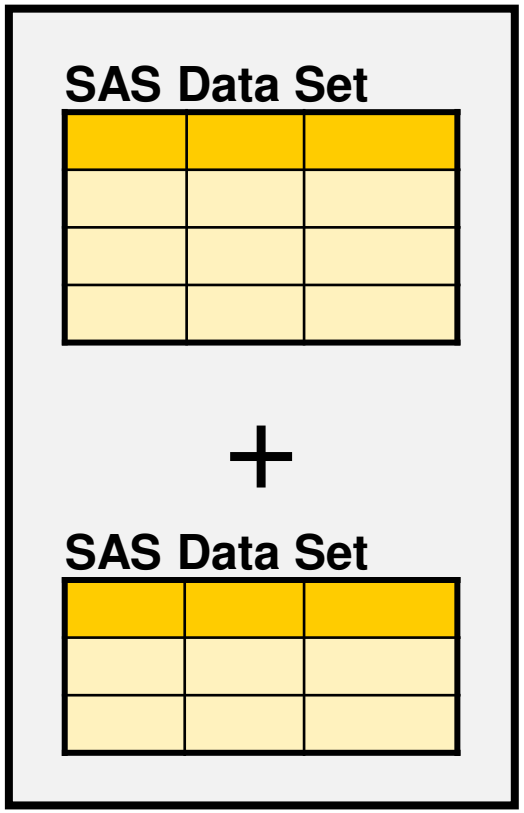


Objectives

- Append one SAS data set to another SAS data set by using the APPEND procedure.
- Append a SAS data set containing additional variables to another SAS data set by using the FORCE option with the APPEND procedure.

Appending and Concatenating

Appending and concatenating involves combining SAS data sets, one after the other, into a single SAS data set.



- Appending adds the observations in the second data set directly to the end of the original data set.
- Concatenating copies all observations from the first data set and then copies all observations from one or more successive data sets into a new data set.

The APPEND Procedure

The *APPEND procedure* adds the observations from one SAS data set to the end of another SAS data set.

General form of the APPEND procedure:

```
PROC APPEND  BASE = SAS-data-set
             DATA = SAS-data-set;
RUN;
```

BASE= names the data set to which observations are added.

DATA= names the data set containing observations that are added to the base data set.

The APPEND Procedure

Requirements:

- Only two data sets can be used at a time in one step.
- The observations in the base data set are not read.
- The variable information in the descriptor portion of the base data set cannot change.

Business Scenario

Emps is a master data set that contains employees hired in 2006 and 2007.

Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007

Business Scenario

Emps is a master data set that contains employees hired in 2006 and 2007.

Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007

The employees hired in 2008, 2009, and 2010 need to be appended.

Emps2008

First	Gender	HireYear
Brett	M	2008
Renee	F	2008

Emps2009

First	HireYear
Sara	2009
Dennis	2009

Emps2010

First	HireYear	Country
Rose	2010	Spain
Eric	2010	Spain

10.02 Quiz

How many observations will be in **Emps** after appending the three data sets?

Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007

Emps2008

First	Gender	HireYear
Brett	M	2008
Renee	F	2008

Emps2009

First	HireYear
Sara	2009
Dennis	2009

Emps2010

First	HireYear	Country
Rose	2010	Spain
Eric	2010	Spain

10.03 Quiz

How many variables will be in **Emps** after appending the three data sets?

Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007

Emps2008

First	Gender	HireYear
Brett	M	2008
Renee	F	2008

Emps2009

First	HireYear
Sara	2009
Dennis	2009

Emps2010

First	HireYear	Country
Rose	2010	Spain
Eric	2010	Spain

Like-Structured Data Sets

Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007

Emps2008

First	Gender	HireYear
Brett	M	2008
Renee	F	2008

The data sets contain the same variables.

```
proc append base=Emps
            data=Emps2008;
run;
```

Like-Structured Data Sets

```
84   proc append base=Emps
85               data=Emps2008;
86   run;

NOTE: Appending WORK.EMPS2008 to WORK.EMPS.
NOTE: There were 2 observations read from the data set
      WORK.EMPS2008.
NOTE: 2 observations added.
NOTE: The data set WORK.EMPS has 5 observations and 3 variables.
```

Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007
Brett	M	2008
Renee	F	2008

Unlike-Structured Data Sets

Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007
Brett	M	2008
Renee	F	2008

Emps2009

First	HireYear
Sara	2009
Dennis	2009

The BASE= data set has a variable that is not in the DATA= data set.

```
proc append base=Emps
            data=Emps2009;
run;
```

Unlike-Structured Data Sets

```
90    proc append base=Emps
91                data=Emps2009;
92    run;

NOTE: Appending WORK.EMPS2009 to WORK.EMPS.
WARNING: Variable Gender was not found on DATA file.
NOTE: There were 2 observations read from the data set
      WORK.EMPS2009.
NOTE: 2 observations added.
NOTE: The data set WORK.EMPS has 7 observations and 3 variables.
```

Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007
Brett	M	2008
Renee	F	2008
Sara		2009
Dennis		2009

Unlike-Structured Data Sets

Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007
Brett	M	2008
Renee	F	2008
Sara		2009
Dennis		2009

Emps2010

First	HireYear	Country
Rose	2010	Spain
Eric	2010	Spain

The DATA= data set has a variable that is not in the BASE= data set.

```
proc append base=Emps
            data=Emps2010;
run;
```

Unlike-Structured Data Sets

```
96    proc append base=Emps
97                data=Emps2010;
98    run;

NOTE: Appending WORK.EMPS2010 to WORK.EMPS.
WARNING: Variable Country was not found on BASE file. The
         variable will not be added to the BASE file.
WARNING: Variable Gender was not found on DATA file.
ERROR: No appending done because of anomalies listed above.
       Use FORCE option to append these files.
NOTE: 0 observations added.
NOTE: The data set WORK.EMPS has 7 observations and 3 variables.
NOTE: Statements not processed because of errors noted above.

NOTE: The SAS System stopped processing this step because of
      errors.
```

Unlike-Structured Data Sets

The *FORCE option* forces the observations to be appended when the DATA= data set contains variables that are not in the BASE= data set.

General form of the FORCE option:

```
PROC APPEND  BASE = SAS-data-set
             DATA = SAS-data-set  FORCE;
RUN;
```

The FORCE option causes the extra variables to be dropped and issues a warning message.

```
proc append base=Emps
            data=Emps2010 force;
run;
```

Unlike-Structured Data Sets

```
100  proc append base=Emps
101              data=Emps2010 force;
102  run;

NOTE: Appending WORK.EMPS2010 to WORK.EMPS.
WARNING: Variable Country was not found on BASE file. The
         variable will not be added to the BASE file.
WARNING: Variable Gender was not found on DATA file.
NOTE: FORCE is specified, so dropping/truncating will occur.
NOTE: There were 2 observations read from the data set
      WORK.EMPS2010.
NOTE: 2 observations added.
NOTE: The data set WORK.EMPS has 9 observations and 3 variables.
```

Unlike-Structured Data Sets

Emps

First	Gender	HireYear
Stacey	F	2006
Gloria	F	2007
James	M	2007
Brett	M	2008
Renee	F	2008
Sara		2009
Dennis		2009
Rose		2010
Eric		2010

Unlike-Structured Data Sets

Situation	Action
BASE= data set contains a variable that is not in the DATA= data set.	The observations are appended, but the observations from the DATA= data set have a missing value for the variable that was not present in the DATA= data set. The FORCE option is not necessary in this case.
DATA= data set contains a variable that is not in the BASE= data set.	Use the FORCE option in the PROC APPEND statement to force the concatenation of the two data sets. The statement drops the extra variable and issues a warning message.

10.04 Quiz

How many observations will be in **Emps** if the program is submitted a second time?

Submitting this program once appends six observations to the **Emps** data set, which results in a total of nine observations.

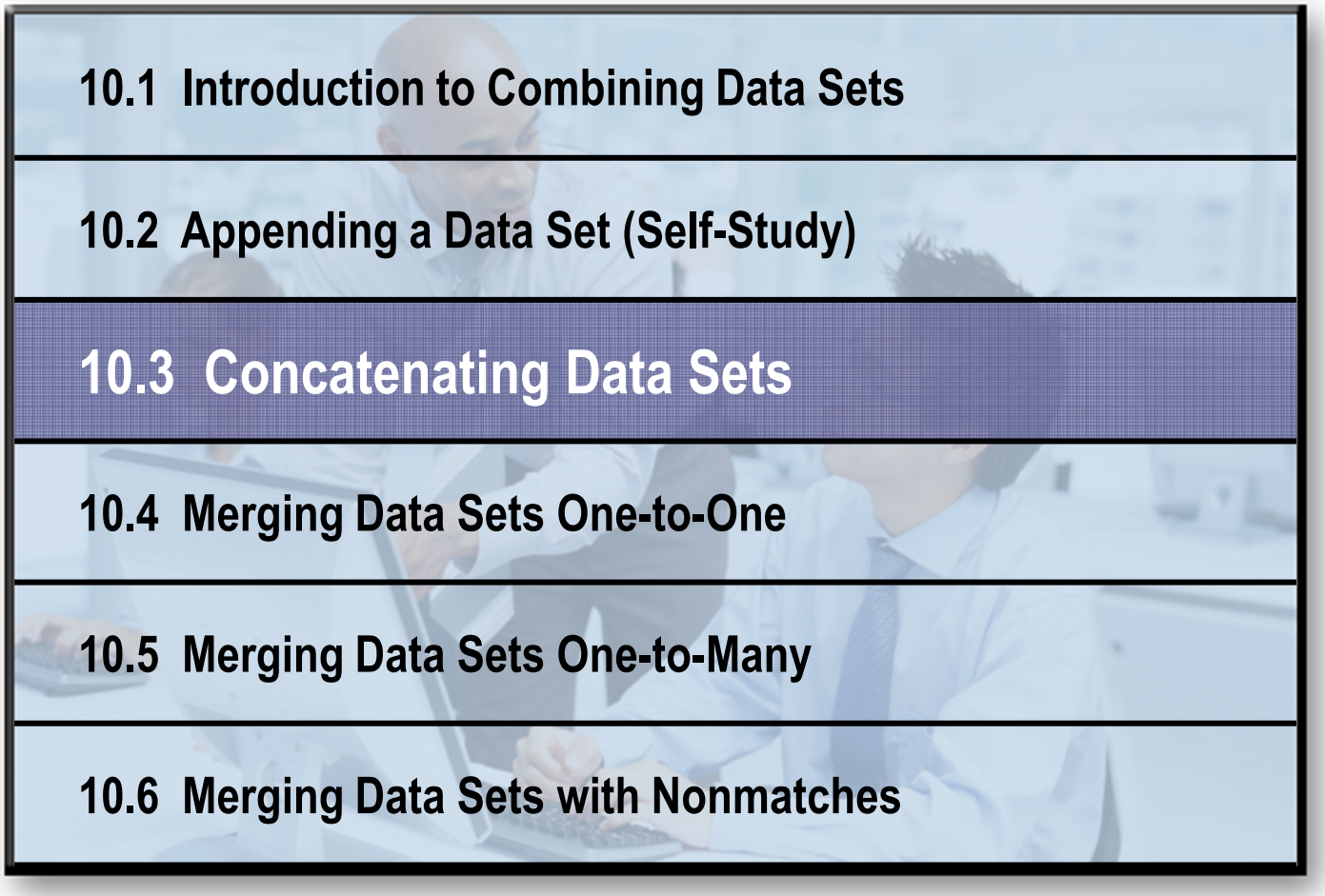
```
proc append base=Emps
            data=Emps2008;
run;
proc append base=Emps
            data=Emps2009;
run;
proc append base=Emps
            data=Emps2010 force;
run;
```

3 obs + 2 obs = 5 obs

5 obs + 2 obs = 7 obs

7 obs + 2 obs = 9 obs

Chapter 10: Combining SAS Data Sets

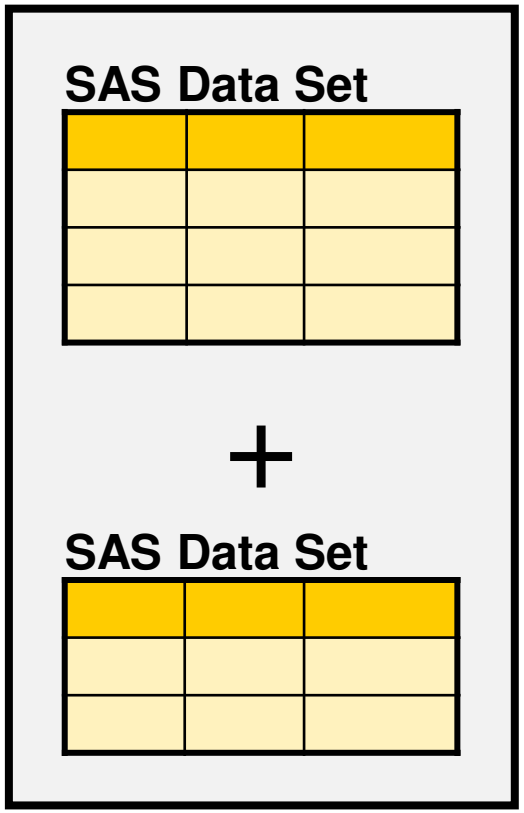


Objectives

- Concatenate two or more SAS data sets by using the SET statement in a DATA step.
- Change the names of variables by using the RENAME= data set option.
- Compare the APPEND procedure to the SET statement. (Self-Study)
- Interleave two or more SAS data sets by using the SET and BY statements in a DATA step. (Self-Study)

Appending and Concatenating

Appending and concatenating involves combining SAS data sets, one after the other, into a single SAS data set.



- Appending adds the observations in the second data set directly to the end of the original data set.

- Concatenating copies all observations from the first data set and then copies all observations from one or more successive data sets into a new data set.

The SET Statement

The *SET statement* in a DATA step reads observations from one or more SAS data sets.

```
DATA SAS-data-set;
      SET SAS-data-set1 SAS-data-set2 . . .;
      <additional SAS statements>
RUN;
```

- Any number of data sets can be in the SET statement.
- The observations from the first data set in the SET statement appear first in the new data set. The observations from the second data set follow those from the first data set, and so on.

Like-Structured Data Sets

Concatenate **EmpsDK** and **EmpsFR** to create a new data set named **EmpsAll1**.

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

The data sets contain the same variables.

```
data EmpsAll1;
   set EmpsDK EmpsFR;
run;
```

Compilation

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
   set EmpsDK EmpsFR;
run;
```

PDV

First	Gender	Country

EmpsAll1

First	Gender	Country

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK
run;
```

Initialize PDV

PDV

First	Gender	Country

EmpsAll1

First	Gender	Country

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

PDV

First	Gender	Country
Lars	M	Denmark

EmpsAll1

First	Gender	Country

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	Country
Lars	M	Denmark

→

First	Gender	Country
Lars	M	Denmark

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

PDV

First	Gender	Country
Kari	F	Denmark

EmpsAll1

First	Gender	Country
Lars	M	Denmark

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	Country
Kari	F	Denmark

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

PDV

First	Gender	Country
Jonas	M	Denmark

EmpsAll1

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	Country
Jonas	M	Denmark

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EOF

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

PDV

First	Gender	Country
Jonas	M	Denmark

EmpsAll1

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK
run;
```

Reinitialize PDV

PDV

First	Gender	Country

EmpsAll1

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

PDV

First	Gender	Country
Pierre	M	France

EmpsAll1

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	Country
Pierre	M	France

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark
Pierre	M	France

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

PDV

First	Gender	Country
Sophie	F	France

EmpsAll1

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark
Pierre	M	France

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	Country
Sophie	F	France

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark
Pierre	M	France
Sophie	F	France

...

Execution

EmpsDK

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark

EmpsFR

First	Gender	Country
Pierre	M	France
Sophie	F	France

EOF

```
data EmpsAll1;
    set EmpsDK EmpsFR;
run;
```

PDV

First	Gender	Country
Sophie	F	France

EmpsAll1

First	Gender	Country
Lars	M	Denmark
Kari	F	Denmark
Jonas	M	Denmark
Pierre	M	France
Sophie	F	France

Unlike-Structured Data Sets

Concatenate **EmpsCN** and **EmpsJP** to create a new data set named **EmpsAll2**.

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

The data sets do not contain the same variables.

```
data EmpsAll2;
   set EmpsCN EmpsJP;
run;
```

10.05 Quiz

How many variables will be in **EmpsAll2** after concatenating **EmpsCN** and **EmpsJP**?

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

```
data EmpsAll2;
   set EmpsCN EmpsJP;
run;
```

Compilation

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

```
data EmpsAll2;
   set EmpsCN EmpsJP;
run;
```

PDV

First	Gender	Country

...

Compilation

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

```
data EmpsAll2;
   set EmpsCN EmpsJP;
run;
```

PDV

First	Gender	Country	Region

Final Results

EmpsAll2

First	Gender	Country	Region
Chang	M	China	
Li	M	China	
Ming	F	China	
Cho	F		Japan
Tomi	M		Japan

The RENAME= Data Set Option

The *RENAME= data set option* changes the name of a variable.

General form of the RENAME= data set option:

SAS-data-set (RENAME = (*old-name-1 = new-name-1*
old-name-2 = new-name-2
...
old-name-n = new-name-n))

- The RENAME= option must be specified in parentheses immediately after the appropriate SAS data set name.
- If the RENAME= option is associated with an input data set in the SET statement, the action applies to the data set that is being read.

The RENAME= Data Set Option

SET statement examples:

```
set  EmpsCN(rename=(Country=Region))
     EmpsJP;
```

```
set  EmpsCN(rename=(First=Fname
                    Country=Region))
     EmpsJP(rename=(First=Fname));
```

```
set  EmpsCN
     EmpsJP(rename=(Region=Country));
```

10.06 Quiz

Which statement has correct syntax?

a.
```
set EmpsCN(rename(Country=Location))
    EmpsJP(rename(Region=Location));
```

b.
```
set EmpsCN(rename=(Country=Location))
    EmpsJP(rename=(Region=Location));
```

c.
```
set EmpsCN rename=(Country=Location)
    EmpsJP rename=(Region=Location);
```

Compilation

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

```
data EmpsAll2;
   set EmpsCN EmpsJP(rename=(Region=Country));
run;
```

PDV

First	Gender	Country

Compilation

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

```
data EmpsAll2;
   set EmpsCN EmpsJP(rename=(Region=Country));
run;
```

PDV

First	Gender	Country

...

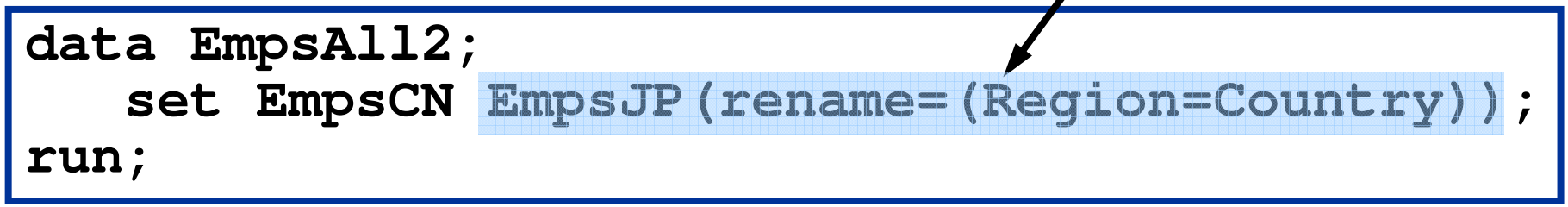
Compilation

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan



```
data EmpsAll2;
   set EmpsCN EmpsJP(rename=(Region=Country));
run;
```

PDV

First	Gender	Country

...

Compilation

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

```
data EmpsAll2;
   set EmpsCN EmpsJP(rename=(Region=Country));
run;
```

PDV

First	Gender	Country

...

Compilation

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

```
data EmpsAll2;
   set EmpsCN EmpsJP(rename=(Region=Country));
run;
```

PDV

First	Gender	Country

Final Results

EmpsAll2

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China
Cho	F	Japan
Tomi	M	Japan

APPEND Procedure versus SET Statement (Self-Study)

- The data set that results from concatenating two data sets with the SET statement is the same data set that results from concatenating them with the APPEND procedure if the two data sets contain the same variables.
- The APPEND procedure concatenates much faster than the SET statement because the APPEND procedure does not process the observations from the BASE= data set.
- The two methods are significantly different when the variables differ between data sets.

APPEND Procedure versus SET Statement (Self-Study)

Criterion	APPEND Procedure	SET Statement
Number of data sets that you can concatenate	Uses two data sets.	Uses any number of data sets.
Handling of data sets that contain different variables	Uses all variables in the BASE= data set and assigns missing values to observations from the DATA= data set where appropriate; cannot include variables found only in the DATA= data set.	Uses all variables and assigns missing values where appropriate.

10.07 Multiple Choice Poll (Self-Study)

Which method would you use if you wanted to create a new variable at the time of concatenation?

a. APPEND procedure
b. SET statement

Interleaving (Self-Study)

Interleaving intersperses observations from two or more data sets, based on one or more common variables.

The SET statement with a BY statement in a DATA step interleaves SAS data sets.

```
DATA SAS-data-set;
      SET SAS-data-set1 SAS-data-set2 . . .;
      BY <DESCENDING> by-variable(s);
      <additional SAS statements>
RUN;
```

The data sets must be sorted by the BY *variable*.

Use the SORT procedure to sort the data sets by the BY variable.

Interleaving (Self-Study)

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

Which value comes first?

Chang

```
data EmpsAll2;
   set EmpsCN EmpsJP(rename=(Region=Country));
   by First;
run;
```

PDV

First	Gender	Country
Chang	M	China

Interleaving (Self-Study)

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

Which value comes first?

Cho

```
data EmpsAll2;
   set EmpsCN
   by First;
run;
```

Reinitialize PDV

```
(Region=Country));
```

PDV

First	Gender	Country

...

Interleaving (Self-Study)

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

Which value comes first?

Cho

```
data EmpsAll2;
   set EmpsCN EmpsJP(rename=(Region=Country));
   by First;
run;
```

PDV

First	Gender	Country
Cho	F	Japan

...

Interleaving (Self-Study)

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

Which value comes first?

Li

```
data EmpsAll2;
   set EmpsCN
   by First;
run;
```

Reinitialize PDV

```
(Region=Country));
```

PDV

First	Gender	Country

...

Interleaving (Self-Study)

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

Which value comes first?

Li

```
data EmpsAll2;
   set EmpsCN EmpsJP(rename=(Region=Country));
   by First;
run;
```

PDV

First	Gender	Country
Li	M	China

...

Interleaving (Self-Study)

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
Ming	F	China

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

Which value comes first?

Ming

```
data EmpsAll2;
   set EmpsCN EmpsJP(rename=(Region=Country));
   by First;
run;
```

PDV

First	Gender	Country
Ming	F	China

...

Interleaving (Self-Study)

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
ng	F	China

EOF

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

Which value comes first?

Tomi

```
data EmpsAll2;
   set EmpsCN
   by First;
run;
```

Reinitialize PDV

```
(Region=Country));
```

PDV

First	Gender	Country

...

Interleaving (Self-Study)

EmpsCN

First	Gender	Country
Chang	M	China
Li	M	China
ng	F	China

EOF

EmpsJP

First	Gender	Region
Cho	F	Japan
Tomi	M	Japan

Which value comes first?

Tomi

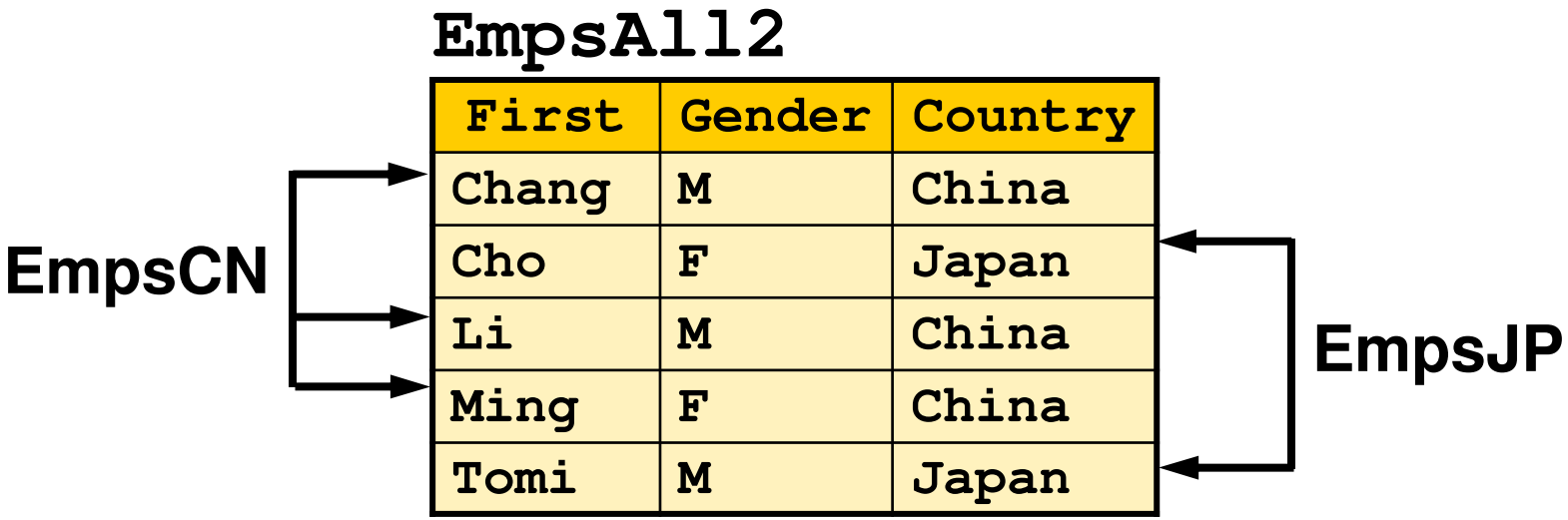
```
data EmpsAll2;
   set EmpsCN EmpsJP(rename=(Region=Country));
   by First;
run;
```

PDV

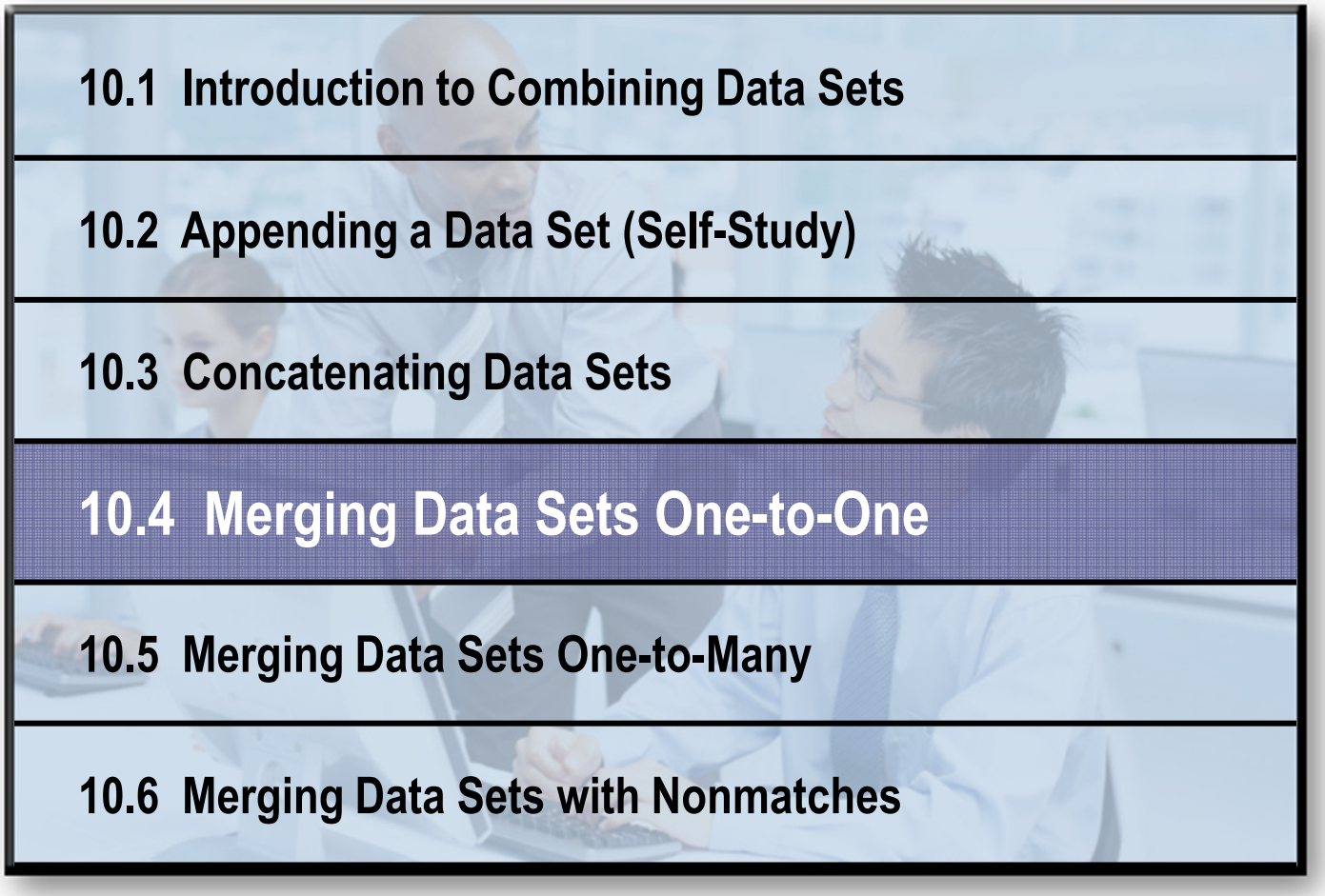
First	Gender	Country
Tomi	M	Japan

...

Interleaving (Self-Study)



Chapter 10: Combining SAS Data Sets

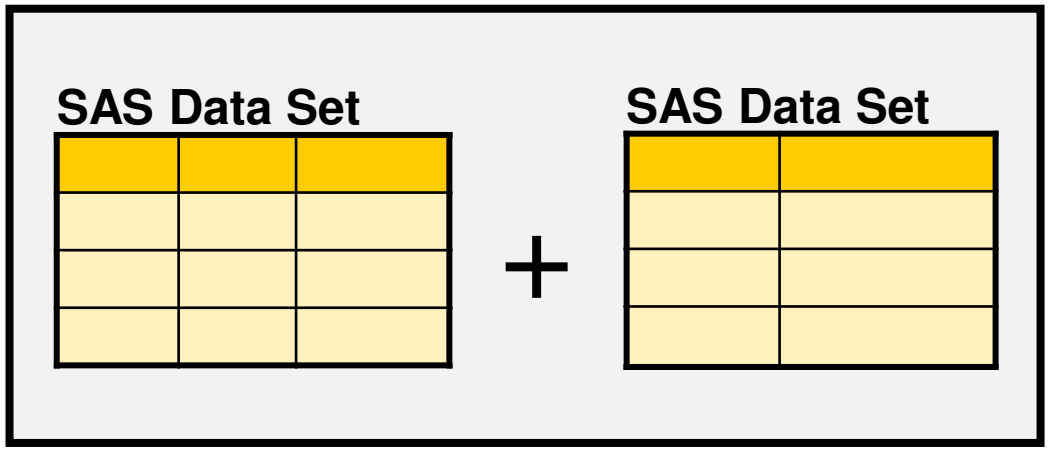


Objectives

- Define the different types of match-merging.
- Prepare data sets for merging using the SORT procedure.
- Merge SAS data sets one-to-one based on a common variable by using the MERGE and BY statements in a DATA step.
- Eliminate duplicate observations using the SORT procedure. (Self-Study)

Merging

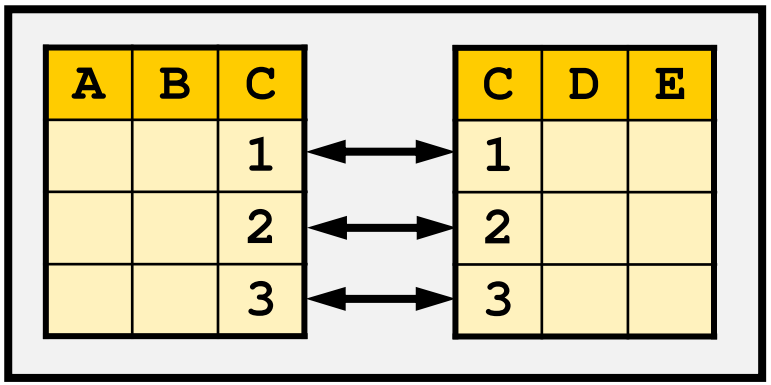
Merging involves combining observations from two or more SAS data sets into a single observation in a new SAS data set.

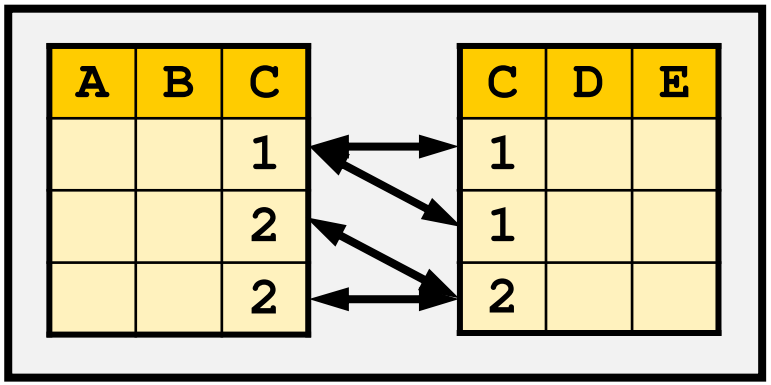


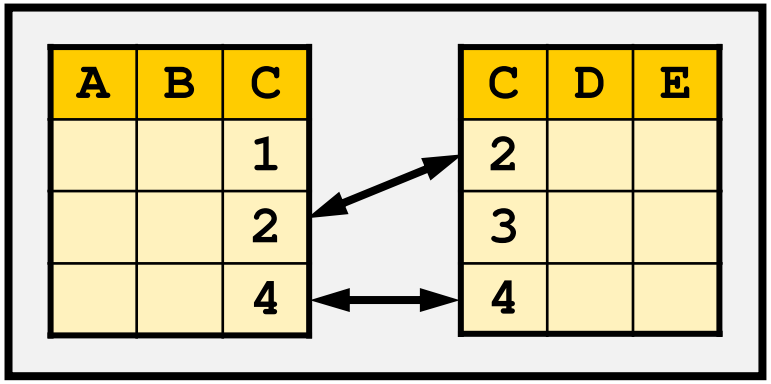
Observations can be merged based on their positions in the original data sets or merged by one or more common variables.

Match-Merging

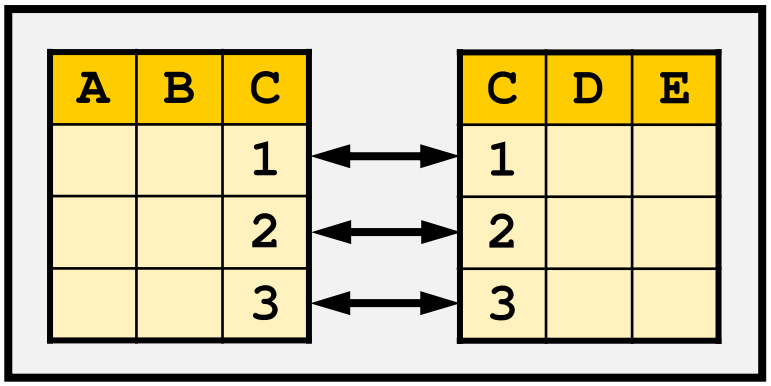
Match-merging combines observations from two or more SAS data sets into a single observation in a new data set based on the values of one or more common variables.





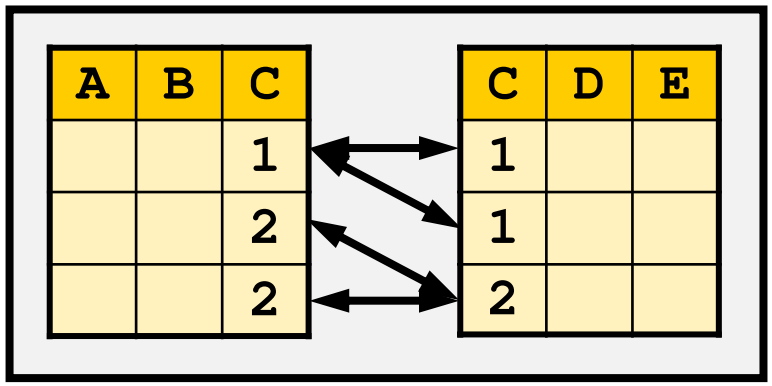


Match-Merging



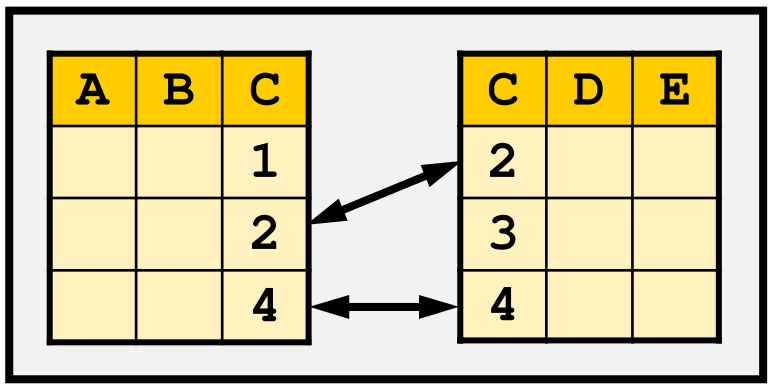
One-to-One

A single observation in one data set is related to one and only one observation from another data set based on the values of one or more selected variables.



One-to-Many or Many-to-One

A single observation in one data set is related to more than one observation from another data set based on the values of one or more selected variables and vice versa.



Nonmatches

At least one single observation in one data set is unrelated to any observation from another data set based on the values of one or more selected variables.

Match-Merging

In order to perform match-merging, the observations in each data set must be sorted by the one or more common variables that are being matched.

General form of the SORT procedure:

```
PROC SORT  DATA=input-SAS-data-set
                <OUT=output-SAS-data-set>;
     BY <DESCENDING> by-variable(s);
RUN;
```

The *SORT procedure* orders SAS data set observations by the values of one or more variables.

The SORT Procedure

```
PROC SORT  DATA=input-SAS-data-set
                <OUT=output-SAS-data-set>;
     BY <DESCENDING> by-variable(s);
RUN;
```

The SORT procedure

- rearranges the observations in a SAS data set
- either replaces the original data set or creates a new data set
- can sort on multiple variables
- can sort in ascending (default) or descending order
- does not generate printed output.

10.08 Quiz

Which step is sorting the observations in a SAS data set and overwriting the same SAS data set?

a.
```
proc sort data=work.EmpsAU
          out=work.sorted;
   by First;
run;
```

b.
```
proc sort data=work.EmpsAU
          out=orion.EmpsAU;
   by First;
run;
```

c.
```
proc sort data=work.EmpsAU;
   by First;
run;
```

The BY Statement

The *BY statement* specifies the sorting variables.

- PROC SORT first arranges the data set by the values in ascending order, by default, of the first BY variable.
- PROC SORT then arranges any observations that have the same value of the first BY variable by the values of the second BY variable in ascending order.
- This sorting continues for every specified BY variable.

The *DESCENDING option* reverses the sort order for the variable that immediately follows in the statement so that observations are sorted from the largest value to the smallest value.

The BY Statement

BY statement examples:

```
by Last First;
```

```
by descending Last First;
```

```
by Last descending First;
```

```
by descending Last descending First;
```

Setup for the Poll

- Retrieve program **p110a01**.
- Add a BY statement to the PROC SORT step to sort the observations first by ascending **Gender** and then by descending **Employee_ID** within the values of **Gender**.
- Complete the PROC PRINT statement to reference the sorted data set.
- Submit the program and confirm the sort order in the PROC PRINT output.

10.09 Multiple Choice Poll

What is the **`Employee_ID`** value for the first observation in the sorted data set?

a. 120102
b. 120121
c. 121144
d. 121145

The MERGE and BY Statements

The *MERGE statement* in a DATA step joins observations from two or more SAS data sets into single observations.

```
DATA SAS-data-set;
      MERGE SAS-data-set1 SAS-data-set2 . . .;
      BY <DESCENDING> by-variable(s);
      <additional SAS statements>
RUN;
```

A *BY statement* after the MERGE statement performs a match-merge.

The MERGE and BY Statements

Requirements when two or more SAS data sets are specified in the MERGE statement:

- The variables in the BY statement must be common to all data sets.
- The data sets that are listed in the MERGE statement must be sorted in the order of the values of the variables that are listed in the BY statement.

One-to-One Merge

Merge **EmpsAU** and **PhoneH** by **EmpID** to create a new data set named **EmpsAUH**.

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneH

EmpID	Phone
121150	+61(2)5555-1793
121151	+61(2)5555-1849
121152	+61(2)5555-1665

The data sets are sorted by **EmpID**.

```
data EmpsAUH;
   merge EmpsAU PhoneH;
   by EmpID;
run;
```

Final Results

EmpsAUH

First	Gender	EmpID	Phone
Togar	M	121150	+61(2)5555-1793
Kylie	F	121151	+61(2)5555-1849
Birin	M	121152	+61(2)5555-1665

10.10 Quiz

- Retrieve program **p110a02**.
- Complete the program to match-merge the sorted SAS data sets referenced in the PROC SORT steps.
- Submit the program. Correct and resubmit, if necessary.

What are the modified, completed statements?

Eliminating Duplicates with the SORT Procedure (Self-Study)

The SORT procedure can be used to eliminate duplicate observations.

PROC SORT Statement Options:

- The *NODUPKEY option* deletes observations with duplicate BY values.
- The *EQUALS option* maintains the relative order of the observations within the input data set in the output data set for observations with identical BY values.

Eliminating Duplicates with the SORT Procedure (Self-Study)

```
proc sort data=EmpsDUP
          out=EmpsDUP1 nodupkey equals;
   by EmpID;
run;
```

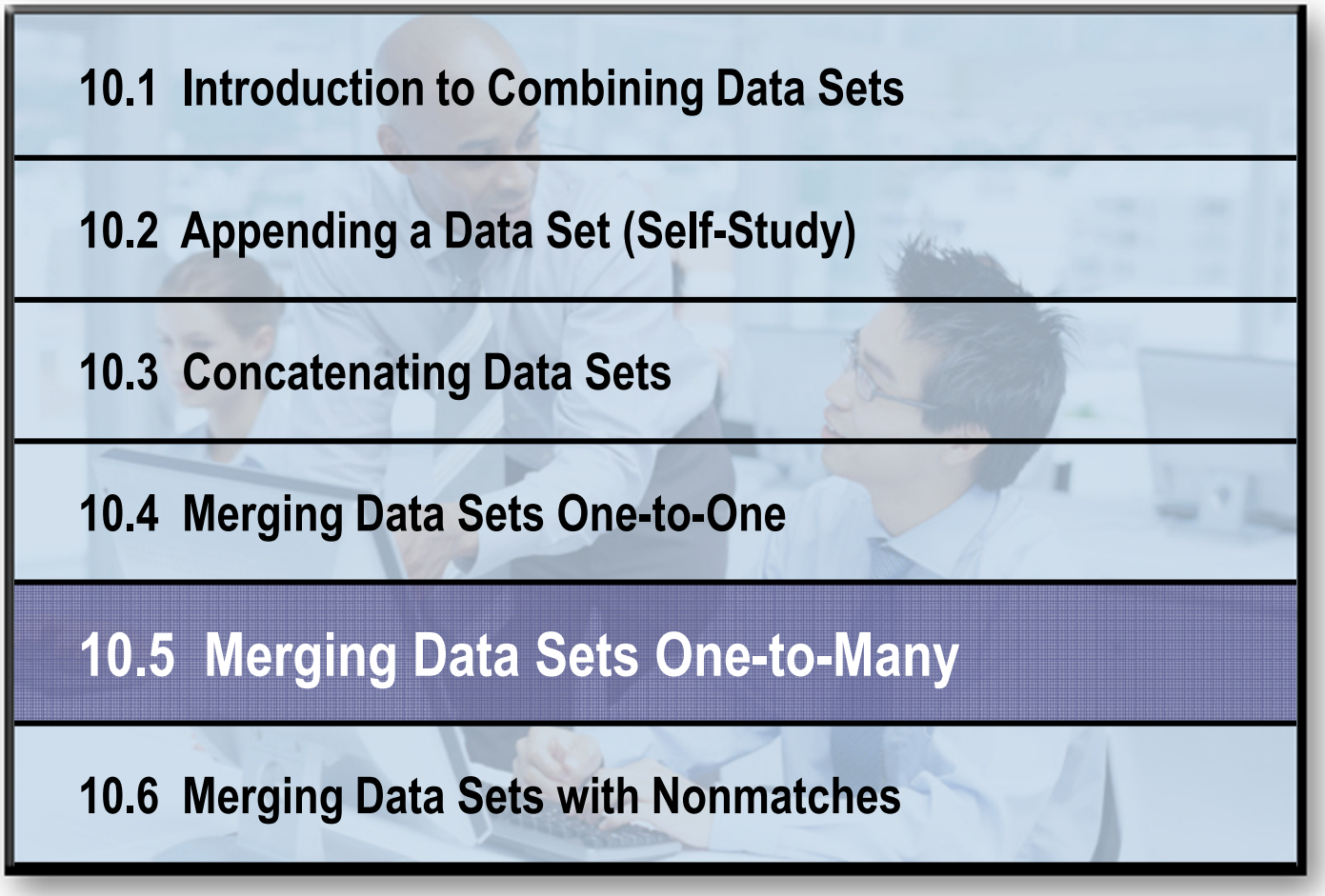
EmpsDUP

First	Gender	EmpID
Matt	M	121160
Julie	F	121161
Brett	M	121162
Julie	F	121161
Chris	F	121161
Julie	F	121163

EmpsDUP1

First	Gender	EmpID
Matt	M	121160
Julie	F	121161
Brett	M	121162
Julie	F	121163

Chapter 10: Combining SAS Data Sets



Objectives

- Merge SAS data sets one-to-many based on a common variable by using the MERGE and BY statements in a DATA step.

One-to-Many Merge

Merge **EmpsAU** and **PhoneHW** by **EmpID** to create a new data set named **EmpsAUHW**.

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
   merge EmpsAU PhoneHW;
   by EmpID;
run;
```

The data sets are sorted by **EmpID**.

p110d06

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU
    by EmpID;
run;
```

Initialize PDV

PDV

First	Gender	EmpID	Type	Phone
		.		

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Do the **EmpID**s match?

Yes

PDV

First	Gender	EmpID	Type	Phone
		.		

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Reads one observation from each matching data set

PDV

First	Gender	EmpID	Type	Phone
Togar	M	121150	Home	+61(2)5555-1793

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	EmpID	Type	Phone
Togar	M	121150	Home	+61(2)5555-1793

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Do the **EmpID**s match?

No

PDV

First	Gender	EmpID	Type	Phone
Togar	M	121150	Home	+61(2)5555-1793

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Is either **EmpID** the same as the **EmpID** currently in the PDV?

Yes

PDV

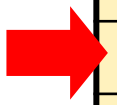
First	Gender	EmpID	Type	Phone
Togar	M	121150	Home	+61(2)5555-1793

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152



PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Reads the observation from the appropriate data set

PDV

First	Gender	EmpID	Type	Phone
Togar	M	121150	Work	+61(2)5555-1794

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	EmpID	Type	Phone
Togar	M	121150	Work	+61(2)5555-1794

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Do the **EmpID**s match?

Yes

PDV

First	Gender	EmpID	Type	Phone
Togar	M	121150	Work	+61(2)5555-1794

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Is the **EmpID** the same as the **EmpID** currently in the PDV?

No

PDV

First	Gender	EmpID	Type	Phone
Togar	M	121150	Work	+61(2)5555-1794

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Reinitialize PDV

PDV

First	Gender	EmpID	Type	Phone
		.		

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Reads one observation from each matching data set

PDV

First	Gender	EmpID	Type	Phone
Kylie	F	121151	Home	+61(2)5555-1849

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	EmpID	Type	Phone
Kylie	F	121151	Home	+61(2)5555-1849

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Do the **EmpID**s match?

No

PDV

First	Gender	EmpID	Type	Phone
Kylie	F	121151	Home	+61(2)5555-1849

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Is either **EmpID** the same as the **EmpID** currently in the PDV?

Yes

PDV

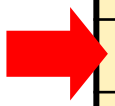
First	Gender	EmpID	Type	Phone
Kylie	F	121151	Home	+61(2)5555-1849

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152



PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Reads the observation from the appropriate data set

PDV

First	Gender	EmpID	Type	Phone
Kylie	F	121151	Work	+61(2)5555-1850

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
    merge EmpsAU PhoneHW;
    by EmpID;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	EmpID	Type	Phone
Kylie	F	121151	Work	+61(2)5555-1850

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneHW

EmpID	Type	Phone
121150	Home	+61(2)5555-1793
121150	Work	+61(2)5555-1794
121151	Home	+61(2)5555-1849
121151	Work	+61(2)5555-1850
121152	Home	+61(2)5555-1665
121152	Work	+61(2)5555-1666

```
data EmpsAUHW;
   merge EmpsAU Ph
   by EmpID;
run;
```

Continue until EOF on both data sets

PDV

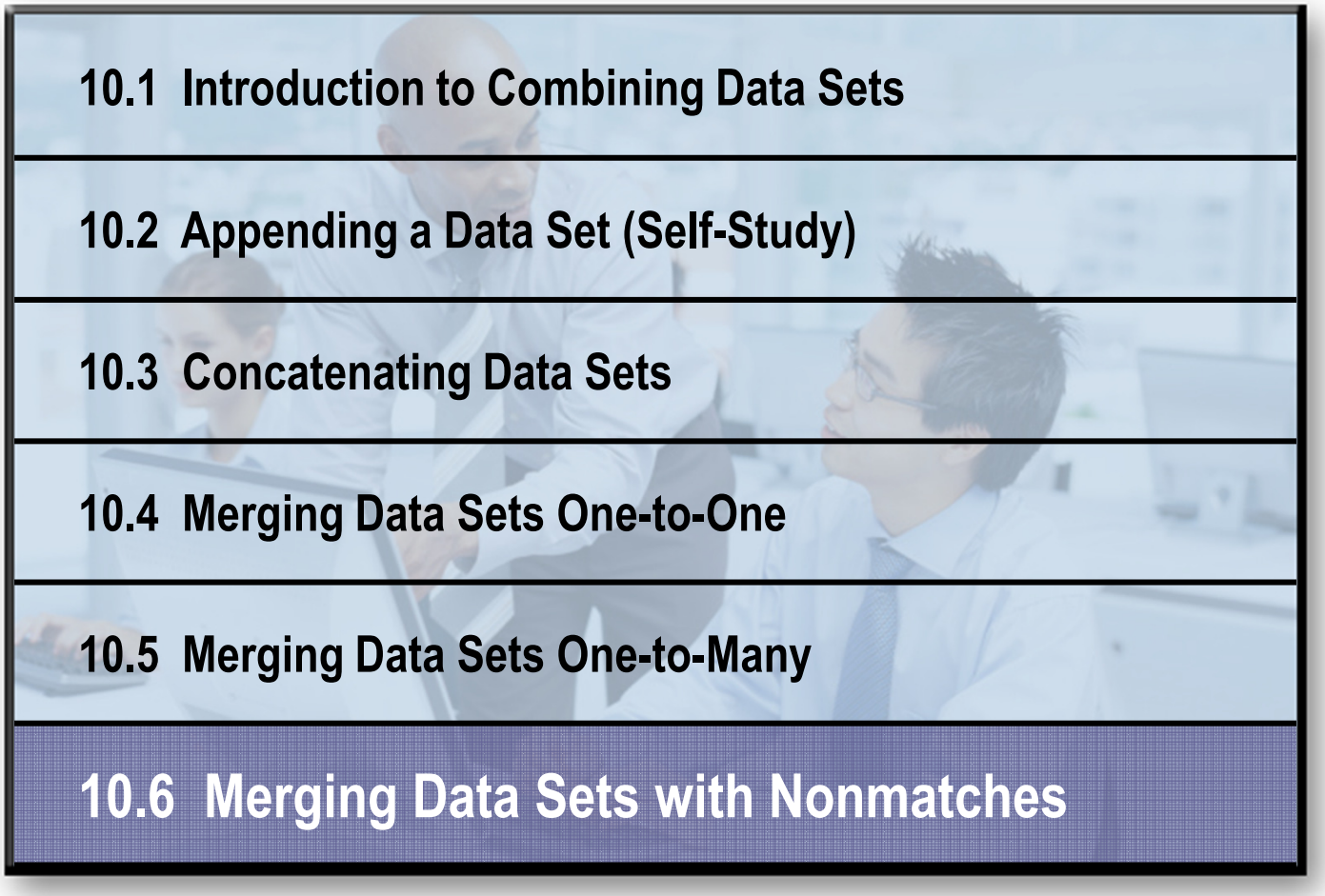
First	Gender	EmpID	Type	Phone
Kylie	F	121151	Work	+61(2)5555-1850

Final Results

EmpsAUHW

First	Gender	EmpID	Type	Phone
Togar	M	121150	Home	+61(2)5555-1793
Togar	M	121150	Work	+61(2)5555-1794
Kylie	F	121151	Home	+61(2)5555-1849
Kylie	F	121151	Work	+61(2)5555-1850
Birin	M	121152	Home	+61(2)5555-1665
Birin	M	121152	Work	+61(2)5555-1666

Chapter 10: Combining SAS Data Sets



Objectives

- Control the observations in the output data set by using the IN= option.
- Output observations to multiple data sets using the IN= option and the OUTPUT statement. (Self-Study)
- Compare the results of a many-to-many merge based on using the DATA step or the SQL procedure. (Self-Study)

Nonmatches Merge

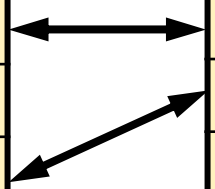
Merge **EmpsAU** and **PhoneC** by **EmpID** to create a new data set named **EmpsAUC**.

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348



The data sets are sorted by **EmpID**.

```
data EmpsAUC;
   merge EmpsAU PhoneC;
   by EmpID;
run;
```

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsA
    by EmpID;
run;
```

Initialize PDV

PDV

First	Gender	EmpID	Phone
		.	

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Do the **EmpID**s match?

Yes

PDV

First	Gender	EmpID	Phone
		.	

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Reads one observation from each matching data set

PDV

First	Gender	EmpID	Phone
Togar	M	121150	+61(2)5555-1795

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
   merge EmpsAU PhoneC;
   by EmpID;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	EmpID	Phone
Togar	M	121150	+61(2)5555-1795

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Do the **EmpID**s match?

No

PDV

First	Gender	EmpID	Phone
Togar	M	121150	+61(2)5555-1795

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Is either **EmpID** the same as the **EmpID** currently in the PDV?

No

PDV

First	Gender	EmpID	Phone
Togar	M	121150	+61(2)5555-1795

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Reinitialize PDV

PDV

First	Gender	EmpID	Phone
		.	

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Which **EmpID** sequentially comes first?

121151

PDV

First	Gender	EmpID	Phone
		.	

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Reads the observation from the **EmpID** that sequentially comes first

PDV

First	Gender	EmpID	Phone
Kylie	F	121151	

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
   merge EmpsAU PhoneC;
   by EmpID;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	EmpID	Phone
Kylie	F	121151	

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Do the **EmpID**s match?

Yes

PDV

First	Gender	EmpID	Phone
Kylie	F	121151	

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Is either **EmpID** the same as the **EmpID** currently in the PDV?

No

PDV

First	Gender	EmpID	Phone
Kylie	F	121151	

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Reinitialize PDV

PDV

First	Gender	EmpID	Phone
		.	

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Reads one observation from each matching data set

PDV

First	Gender	EmpID	Phone
Birin	M	121152	+61(2)5555-1667

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
   merge EmpsAU PhoneC;
   by EmpID;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	EmpID	Phone
Birin	M	121152	+61(2)5555-1667

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

EOF

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Is the **EmpID** the same as the **EmpID** currently in the PDV?

No

PDV

First	Gender	EmpID	Phone
Birin	M	121152	+61(2)5555-1667

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
n	M	121152

EOF

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Reinitialize PDV

PDV

First	Gender	EmpID	Phone
		.	

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
n	M	121152

EOF

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

Reads the observation from the appropriate data set

PDV

First	Gender	EmpID	Phone
		121153	+61(2)5555-1348

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
n	M	121152

EOF

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
   merge EmpsAU PhoneC;
   by EmpID;
run;
```

Implicit OUTPUT;
Implicit RETURN;

PDV

First	Gender	EmpID	Phone
		121153	+61(2)5555-1348

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
[illegible]n	M	121152

EOF

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
[illegible]53	+61(2)5555-1348

EOF

```
data EmpsAUC;
    merge EmpsAU PhoneC;
    by EmpID;
run;
```

PDV

First	Gender	EmpID	Phone
		121153	+61(2)5555-1348

Final Results

EmpsAUC

First	Gender	EmpID	Phone
Togar	M	121150	+61(2)5555-1795
Kylie	F	121151	
Birin	M	121152	+61(2)5555-1667
		121153	+61(2)5555-1348

The final results include matches and nonmatches.

- Matches are observations that contain data from both input data sets.
- Nonmatches are observations that contain data from only one input data set.

10.11 Quiz

How many observations in the final data set **EmpsAUC** are considered nonmatches?

a. 1
b. 2
c. 3
d. 4

EmpsAUC

First	Gender	EmpID	Phone
Togar	M	121150	+61(2)5555-1795
Kylie	F	121151	
Birin	M	121152	+61(2)5555-1667
		121153	+61(2)5555-1348

The IN= Data Set Option

The *IN= data set option* creates a variable that indicates whether the data set contributed data to the current observation.

General form of the IN= data set option:

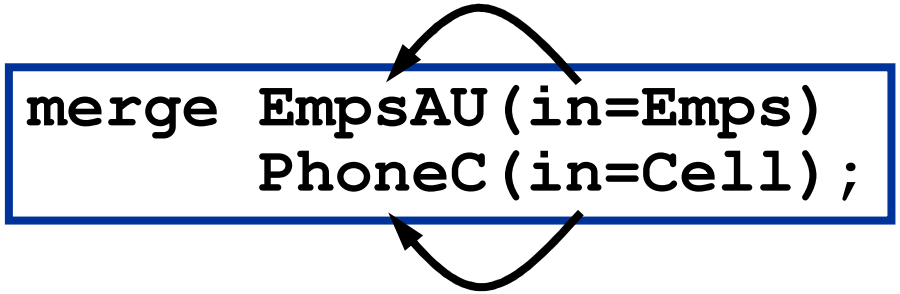
SAS-data-set (IN = *variable*)

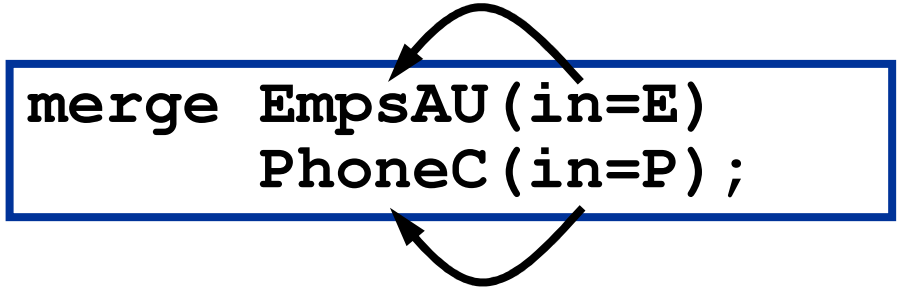
variable is a temporary numeric variable that has two possible values:

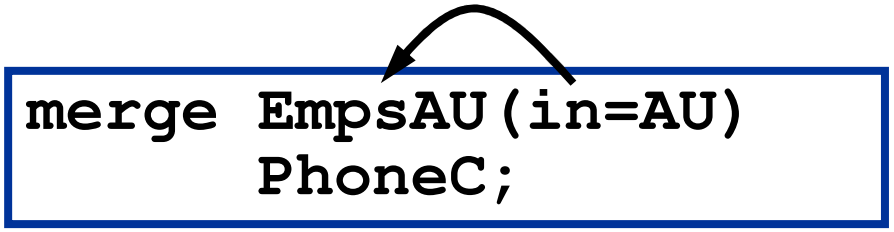
0	indicates that the data set did **not** contribute to the current observation.
1	indicates that the data set **did** contribute to the current observation.

The IN= Data Set Option

MERGE statement examples:







Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
   merge EmpsAU(in=Emps)
         PhoneC(in=Cell);
   by EmpID;
run;
```

PDV

First	Gender	EmpID	D	Emps	Phone	D	Cell
Togar	M	121150		1	+61(2)5555-1795		1

p110d07

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
   merge EmpsAU(in=Emps)
         PhoneC(in=Cell);
   by EmpID;
run;
```

PDV

First	Gender	EmpID	D	Emps	Phone	D	Cell
Kylie	F	121151		1			0

...

Execution

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
   merge EmpsAU(in=Emps)
         PhoneC(in=Cell);
   by EmpID;
run;
```

PDV

First	Gender	EmpID	D	Emps	Phone	D	Cell
Birin	M	121152		1	+61(2)5555-1667		1

...

10.12 Quiz

What are the values of **Emps** and **Cell**?

EmpsAU

First	Gender	EmpID
Togar	M	121150
Kylie	F	121151
Birin	M	121152

PhoneC

EmpID	Phone
121150	+61(2)5555-1795
121152	+61(2)5555-1667
121153	+61(2)5555-1348

```
data EmpsAUC;
   merge EmpsAU(in=Emps)
         PhoneC(in=Cell);
   by EmpID;
run;
```

PDV

First	Gender	EmpID	D	Emps	Phone	D	Cell
		121153			+61(2)5555-1348		

PDV Results

PDV

First	Gender	EmpID	D	Emps	Phone	D	Cell
Togar	M	121150		1	+61(2)5555-1795		1
Kylie	F	121151		1			0
Birin	M	121152		1	+61(2)5555-1667		1
		121153		0	+61(2)5555-1348		1

The variables created with the IN= data set option are only available during execution and are not written to the SAS data set.

10.13 Quiz

Which subsetting IF statement can be added to the DATA step to only output the matches?

a. `if Emps=1 and Cell=0;`

b. `if Emps=1 and Cell=1;`

c. `if Emps=1;`

d. `if Cell=0;`

PDV

First	Gender	EmpID	D Emps	Phone	D Cell
Togar	M	121150	1	+61(2)5555-1795	1
Kylie	F	121151	1		0
Birin	M	121152	1	+61(2)5555-1667	1
		121153	0	+61(2)5555-1348	1

Matches Only

```
data EmpsAUC;
    merge EmpsAU(in=Emps)
          PhoneC(in=Cell);
    by EmpID;
    if Emps=1 and Cell=1;
run;
```

EmpsAUC

First	Gender	EmpID	Phone
Togar	M	121150	+61(2)5555-1795
Birin	M	121152	+61(2)5555-1667

Nonmatches from EmpsAU Only

```
data EmpsAUC;
    merge EmpsAU(in=Emps)
          PhoneC(in=Cell);
    by EmpID;
    if Emps=1 and Cell=0;
run;
```

EmpsAUC

First	Gender	EmpID	Phone
Kylie	F	121151	

Nonmatches from PhoneC Only

```
data EmpsAUC;
    merge EmpsAU(in=Emps)
          PhoneC(in=Cell);
    by EmpID;
    if Emps=0 and Cell=1;
run;
```

EmpsAUC

First	Gender	EmpID	Phone
		121153	+61(2)5555-1348

All Nonmatches

```
data EmpsAUC;
   merge EmpsAU(in=Emps)
         PhoneC(in=Cell);
   by EmpID;
   if Emps=0 or Cell=0;
run;
```

EmpsAUC

First	Gender	EmpID	Phone
Kylie	F	121151	
		121153	+61(2)5555-1348

10.14 Quiz

Write an appropriate IF statement to create the desired data sets.

dataA

X	Y	Z
1	10	20
3	30	40

dataB

X	W
1	50
2	60

```
data new;
   merge dataA(in=A)
         dataB(in=B);
   by X;
run;
```

new

X	Y	Z	W
1	10	20	50
2			60
3	30	40	

Desired SAS Data Sets

X	Y	Z	W
3	30	40	

```
if A=1 and B=0;
        OR
if A and not B;
```

X	Y	Z	W
2			60

X	Y	Z	W
1	10	20	50
3	30	40	

X	Y	Z	W
1	10	20	50
2			60

X	Y	Z	W
1	10	20	50

X	Y	Z	W
2			60
3	30	40	

Outputting to Multiple Data Sets (Self-Study)

The DATA statement can specify multiple output data sets.

```
data EmpsAUC EmpsOnly PhoneOnly;
   merge EmpsAU(in=Emps) PhoneC(in=Cell);
   by EmpID;
   if Emps=1 and Cell=1
         then output EmpsAUC;
   else if Emps=1 and Cell=0
         then output EmpsOnly;
   else if Emps=0 and Cell=1
         then output PhoneOnly;
run;
```

Outputting to Multiple Data Sets (Self-Study)

An OUTPUT statement can be used in a conditional statement to write the current observation to a specific data set that is listed in the DATA statement.

```
data EmpsAUC EmpsOnly PhoneOnly;
   merge EmpsAU(in=Emps) PhoneC(in=Cell);
   by EmpID;
   if Emps=1 and Cell=1
         then output EmpsAUC;
   else if Emps=1 and Cell=0
         then output EmpsOnly;
   else if Emps=0 and Cell=1
         then output PhoneOnly;
run;
```

Outputting to Multiple Data Sets (Self-Study)

EmpsAUC

First	Gender	EmpID	Phone
Togar	M	121150	+61(2)5555-1795
Birin	M	121152	+61(2)5555-1667

EmpsOnly

First	Gender	EmpID	Phone
Kylie	F	121151	

PhoneOnly

First	Gender	EmpID	Phone
		121153	+61(2)5555-1348

Many-to-Many Merge (Self-Study)

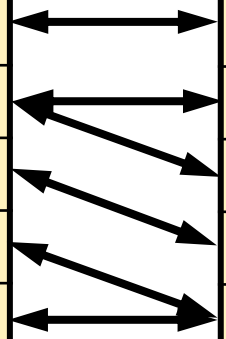
Merge **EmpsAUUS** and **PhoneO** by **Country** to create a new data set named **EmpsOfc**.

EmpsAUUS

First	Gender	Country
Togar	M	AU
Kylie	F	AU
Stacey	F	US
Gloria	F	US
James	M	US

PhoneO

Country	Phone
AU	+61(2)5555-1500
AU	+61(2)5555-1600
AU	+61(2)5555-1700
US	+1(305)555-1500
US	+1(305)555-1600



```
data EmpsOfc;
   merge EmpsAUUS PhoneO;
   by Country;
run;
```

The data sets are sorted by **Country**.

Many-to-Many Merge (Self-Study)

DATA Step Results:

EmpsOfc

First	Gender	Country	Phone
Togar	M	AU	+61(2)5555-1500
Kylie	F	AU	+61(2)5555-1600
Kylie	F	AU	+61(2)5555-1700
Stacey	F	US	+1(305)555-1500
Gloria	F	US	+1(305)555-1600
James	M	US	+1(305)555-1600

Many-to-Many Merge (Self-Study)

The SQL procedure creates different results than the DATA step for a many-to-many merge.

EmpsAUUS

First	Gender	Country
Togar	M	AU
Kylie	F	AU
Stacey	F	US
Gloria	F	US
James	M	US

PhoneO

Country	Phone
AU	+61(2)5555-1500
AU	+61(2)5555-1600
AU	+61(2)5555-1700
US	+1(305)555-1500
US	+1(305)555-1600

```
proc sql;
   create table EmpsOfc as
   select First, Gender, PhoneO.Country, Phone
   from EmpsAUUS, PhoneO
   where EmpsAUUS.Country=PhoneO.Country;
```

Many-to-Many Merge (Self-Study)

PROC SQL Results:

EmpsOfc

First	Gender	Country	Phone
Togar	M	AU	+61(2)5555-1500
Togar	M	AU	+61(2)5555-1600
Togar	M	AU	+61(2)5555-1700
Kylie	F	AU	+61(2)5555-1500
Kylie	F	AU	+61(2)5555-1600
Kylie	F	AU	+61(2)5555-1700
Stacey	F	US	+1(305)555-1500
Stacey	F	US	+1(305)555-1600
Gloria	F	US	+1(305)555-1500
Gloria	F	US	+1(305)555-1600
James	M	US	+1(305)555-1500
James	M	US	+1(305)555-1600

Chapter Review

1. What are the three methods for combining SAS data sets?

2. What data set option enables you to change the name of a variable?

3. What is a requirement of the input SAS data sets prior to match-merging?

4. Which three statements must be used in a DATA step to perform a match-merge?

continued...

Chapter Review

5. Which data set option can be used to prevent non-matches from being written to the output data sets in a match-merge?