STA 311 Statistical Computing & Data Management

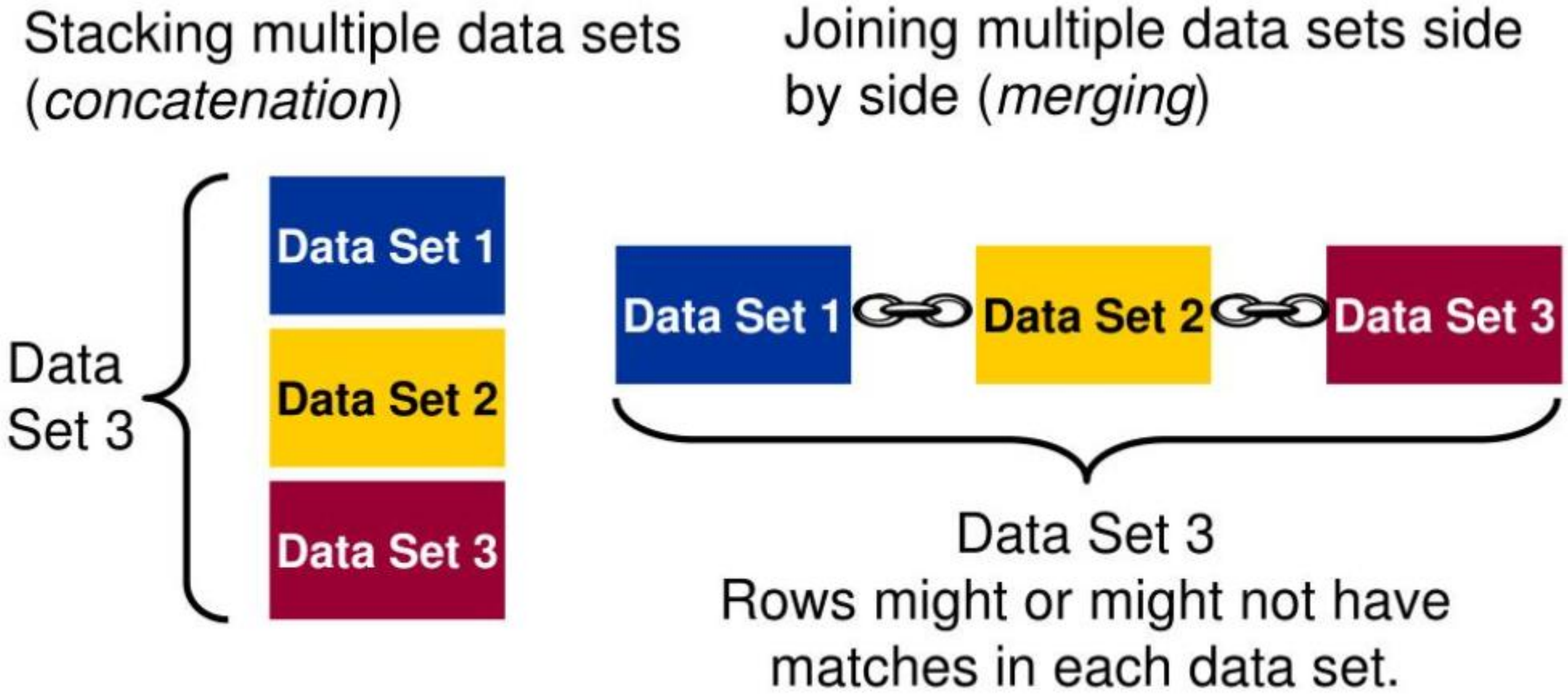
Instructor: Cheng Peng, Ph.D.
Department of Mathematics
West Chester University
West Chester, PA 19383

Office: 25 University Avenue, RM 111
Phone: 610.436.2369
Email: cpeng@wcupa.edu

Topic 8. Combining SAS Data Sets

WCU
WEST CHESTER UNIVERSITY

Topics for This Week



Stacking multiple data sets (*concatenation*)

Joining multiple data sets side by side (*merging*)

Data Set 3
Rows might or might not have matches in each data set.

Topics for This Week

- ❑ One-to-one Reading
  - ○ one-to-one processing – using with caution
- ❑ Concatenating
  - ○ stacking with same variable names and types
  - ○ Data Step options: RENAME, DROP and KEEK
- ❑ Interleaving
  - ○ BY statement with PROC SORT
- ❑ Match Merging
  - ○ BY statement with Sorting
  - ○ IN option – Conditioning with “ghost” variables

One-to-One Reading

When performing one-to-one reading, the new data set contains all the variables from all the input data sets. If the data sets contain same-named variables, the values that are read in from the last data set replace those that were read in from earlier ones. The number of observations in the new data set is the number of observations in the smallest original data set.

```
DATA one2one;
     SET dataset_a;
     SET dataset_b;
RUN;
```

Concatenating Data Sets

To append the observations from one data set to another data set, we concatenate them by specifying the data set names in the SET statement. When SAS concatenates, data sets in the SET statement are read sequentially, in the order in which they are listed. The new data set contains all the variables and the total number of observations from all input data sets.

```
 DATA concat;
    SET dataset_a dataset_b;
RUN;
```

Interleaving Data Sets

If we use a BY statement when you concatenate data sets, the result is interleaving. Interleaving intersperses observations from two or more data sets, based on one or more common variables. Each input data set must be sorted or indexed in ascending order based on the BY variable(s). Observations in each BY group in each data set in the SET statement are read sequentially, in the order in which the data sets and BY variables are listed, until all observations have been processed. The new data set contains all the variables and the total number of observations from all input data sets.

```
DATA interlv;
    SET dataset_a dataset_b;
    BY num;
  RUN;
```

Match-Merge Data Sets

When we combine observations from two or more data sets into a single observation in a new data set according to the values of a same-named variable. This is match-merging, which uses a MERGE statement rather than a SET statement to combine data sets. Each input data set must be sorted or indexed in ascending order based on the BY variable(s). During match-merging, SAS sequentially checks each observation of each data set to see whether the BY values match, then writes the combined observation to the new data set.

```
DATA merged;
 MERGE dataset_a dataset_b;
        BY num;
RUN;
```

How SAS Processes Match-Merge

COMPILING

- ❑ reads the descriptor portions (variable name, type, length) of the data sets that are listed in the MERGE statement
- ❑ reads the rest of the DATA step program
- ❑ creates the program data vector (PDV), an area of memory where SAS builds your data set **one observation at a time**
- ❑ assigns a tracking pointer to each data set that is listed in the MERGE statement.

If variables with the same name appear in more than one data set, the variable from the first data set that contains the variable (in the order listed in the MERGE statement) determines the length of the variable.

How SAS Processes Match-Merge

Executing

After compiling the DATA step, SAS sequentially match-merges observations by moving the pointers down each observation of each data set and checking to see **whether the BY values match**.

- ❑ If **Yes**, the observations are written to the PDV in the order in which the data sets appear in the MERGE statement. Values of any same-named variable are overwritten by values of the same-named variable in subsequent data sets. SAS writes the combined observation to the new data set and retains the values in the PDV until the BY value changes in all the data sets.
- ❑ If **No**, SAS determines which of the values comes first and writes the observation that contains this value to the PDV. Then the observation is written to the new data set.

When the BY value changes in all the input data sets, the PDV is initialized to missing. The DATA step merge continues to process every observation in each data set until it has processed all observations in all data sets.

How SAS Processes Match-Merge

Handling Unmatched Observations and Missing Values

All observations that are written to the PDV, including observations that have missing data and no matching BY values, are written to the output data set.

- ❑ If an observation contains missing values for a variable, then the observation in the output data set contains the missing values as well. Observations that have missing values for the BY variable appear at the top of the output data set.

- ❑ If an input data set doesn't have a matching BY value, then the observation in the output data set contains missing values for the variables that are unique to that input data set.

Some Data Step Options

Renaming Variables

Sometimes you might have same-named variables in more than one input data set. In this case, match-merging overwrites values of the same-named variable in the first data set with values of the same-named variable in subsequent data sets.

To prevent overwriting, use the RENAME= data set option in the MERGE statement to rename variables

Some Data Step Options

Excluding Unmatched Observations

By default, match-merging combines all observations in all input data sets. However, you might want to select only observations that match for two or more input data sets.

To exclude *unmatched observations*, use the IN= data set option and the subsetting IF statement in your DATA step. The IN= data set option creates a variable (‘ghost’ variable because you cannot see it ☺) to indicate whether the data set contributed data to the current observation.

The subsetting IF statement then checks the IN= values and writes to the merged data set only observations that appear in the data sets for which IN= is specified.

Some Data Step Options

Selecting Variables

You can specify the variables you want to drop or keep by using the **DROP =** and **KEEP=** data set options.

When match-merging, you can specify these options in either the DATA statement or the MERGE statement, depending on whether or not you want to process values of the variables in that DATA step.

When used in the DATA statement, the DROP= option simply drops the variables from the new data set. However, they are still read from the original data set and are available within the DATA step.

Some Sample Programs

```
data clinic.one2one;
   set clinic.patients;
      if age<60;
   set clinic.measure;
run;
```

```
data clinic.concat;
   set clinic.therapy1999
       clinic.therapy2000;
run;
```

```
data clinic.intrleav;
   set clinic.therapy1999
       clinic.therapy2000;
   by month;
run;
```

```
data clinic.merged(drop=id);
   merge clinic.demog(in=indemog
                     rename=(date=BirthDate))
         clinic.visit(drop=weight in=invisit
                      rename=(date=VisitDate));
   by id;
   if indemog and invisit;
run;
```

Some Takeaways

❑ You can rename any number of variables in each occurrence of the RENAME= option.

❑ In match-merging, the IN= data set option can apply to any data set in the MERGE statement. The RENAME=, DROP=, and KEEP= options can apply to any data set in the DATA or MERGE statements.

❑ Use the KEEP= option instead of the DROP= option if more variables are dropped than kept.

❑ When you specify multiple data set options for a particular data set, enclose them in a single set of parentheses.