Development Data Boot Camp: Relational Databases: Merge and Append

Ge Sun

University of Notre Dame

May 23, 2023

Introduction

- So far, we have been working with variables from the same dataset. But sometimes, we need to combine variables from different datasets for our analysis.
  * To explore the wage rigidity, we need to combine the national unemployment rate with personal income data
  * We want to explore some long-term patterns, and thus we need to combine census data from different decades.

Introduction

- So far, we have been working with variables from the same dataset. But sometimes, we need to combine variables from different datasets for our analysis.
  - To explore the wage rigidity, we need to combine the national unemployment rate with personal income data
  - We want to explore some long-term patterns, and thus we need to combine census data from different decades.
- To combine two datasets into a new dataset,we use two commands: *merge* and *append*.

Command: *merge*

merge

* The *merge* command combines two datasets to create a wider dataset.
* Typical examples include merging data on the same individuals gathered from two distinct sources and incorporating data on supplementary variables.

	kidname	age	weight
1	Beth	9	60
2	Bob	6	40
3	Barb	3	20
4	Andy	8	80
5	Al	6	50
6	Ann	2	20
7	Pete	6	60
8	Pam	4	40
9	Phil	2	20

	kidname	height
1	Beth	3.4
2	Bob	3.8
3	Barb	2.9
4	Andy	3.2
5	Al	2.8
6	Ann	2.4
7	Pete	3.3
8	Pam	3.4
9	Phil	2.9

Figure 1: Merge example for supplementary variables

Command: *merge*

- Merging two datasets involves adding information from a dataset on disk (not opened in Stata) to a dataset in memory (already opened in Stata).
- The dataset in memory is known as the master dataset, and the dataset on disk can be called using datasets.
- Matching is based on one or more variables called identifiers, such as ID.
- The identifier variables must exist in both data sets, and have the **exact** same names.
- In most cases all *other* variables should have different names.

Question:

	kidname	age	weight
1	Beth	9	60
2	Bob	6	40
3	Barb	3	20
4	Andy	8	80
5	Al	6	50
6	Ann	2	20
7	Pete	6	60
8	Pam	4	40
9	Phil	2	20

	kidname	height
1	Beth	3.4
2	Bob	3.8
3	Barb	2.9
4	Andy	3.2
5	Al	2.8
6	Ann	2.4
7	Pete	3.3
8	Pam	3.4
9	Phil	2.9

Could you tell me if I want to merge the second dataset to the first dataset:

- which one is the identifier?
- which dataset is the master dateset? which dataset is the using dataset?
- what would be the final dataset look like?

Command: *merge*

- Three commonly used merging patterns:
  * 1 : 1 merge: one sample in the master dataset will be matched with one sample in the using dataset. If we've specified that this is a 1:1 merge, the identifier variables must **uniquely identify** observations in both datasets.
  * 1 : m merge: one sample in the master dataset will be matched with many sample in the using dataset. (Consider the case where master dataset contains family-level data while using dataset contains individual-level data). If we've specified that this is a 1:m merge, the identifier variables must uniquely identify observations in the master dataset.
  * m : 1 merge: is essentially as same as 1 : m merge.

1:1 merge

- Basic syntax of *merge*:

  merge 1:1 list_identifiers using filename_using_dataset, ...
  gen (var_merge_results)

- In our example:

  use merge_base.dta,clear
  merge 1:1 kidname using merge_2.dta

```
. merge 1:1 kidname using merge_2.dta

    Result                           # of obs.
    -----------------------------------------
    not matched                             0
    matched                                 9  (_merge==3)
    -----------------------------------------
```

1:1 merge

	kidname	age	weight
1	Beth	9	60
2	Bob	6	40
3	Barb	3	20
4	Andy	8	80
5	Al	6	50
6	Ann	2	20
7	Pete	6	60
8	Pam	4	40
9	Phil	2	20

	kidname	height
1	Beth	3.4
2	Bob	3.8
3	Barb	2.9
4	Andy	3.2
5	Al	2.8
6	Ann	2.4
7	Pete	3.3
8	Pam	3.4
9	Phil	2.9
10	Beth	4

```
use merge_base.dta,clear
merge 1:1 kidname using merge_3.dta
```

1:1 merge

	kidname	age	weight
1	Beth	9	60
2	Bob	6	40
3	Barb	3	20
4	Andy	8	80
5	Al	6	50
6	Ann	2	20
7	Pete	6	60
8	Pam	4	40
9	Phil	2	20

	kidname	height
1	Beth	3.4
2	Bob	3.8
3	Barb	2.9
4	Andy	3.2
5	Al	2.8
6	Ann	2.4
7	Pete	3.3
8	Pam	3.4
9	Phil	2.9
10	Beth	4

use merge_base.dta,clear

merge 1:1 kidname using merge_3.dta

```
. use merge_base.dta,clear

. merge 1:1 kidname using merge_3.dta
variable kidname does not uniquely identify observations in the using data
r(459);

end of do-file

r(459);
```

1:1 merge

How would we find the non-uniquely identified ID?

- *sort*
- *tab*
- *duplicates report kidname*

use merge_3.dta,clear
duplicates report kidname

```
. duplicates report kidname

Duplicates in terms of kidname

   copies | observations       surplus
----------+-------------------------
        1 |            8             0
        2 |            2             1
----------+-------------------------
```

- Also, *duplicates drop* is a good way to drop the same extra rows (Only if you are sure it repeats by mistake!)

1:1 merge

	kidname	age	weight
1	Beth	9	60
2	Bob	6	40
3	Barb	3	20
4	Andy	8	80
5	Al	6	50
6	Ann	2	20
7	Pete	6	60
8	Pam	4	40
9	Phil	2	20

	kidname	height
1	Beth	3.4
2	Bob	3.8
3	Barb	2.9
4	Andy	3.2
5	Al	2.8
6	Ann	2.4
7	Pete	3.3
8	Pam	3.4
9	Phil	2.9
10	Alice	3.1

use merge_base.dta,clear

merge 1:1 kidname using merge_4.dta

```
. merge 1:1 kidname using merge_3.dta
(note: variable kidname was str4, now str5 to accommodate using data's values)

    Result                           # of obs.
    -----------------------------------------
    not matched                             1
        from master                         0  (_merge==1)
        from using                          1  (_merge==2)

    matched                                 9  (_merge==3)
    -----------------------------------------
```

1:1 merge

	kidname	age	weight	height	_merge
1	Al	6	50	2.8	matched (3)
2	Andy	8	80	3.2	matched (3)
3	Ann	2	20	2.4	matched (3)
4	Barb	3	20	2.9	matched (3)
5	Beth	9	60	3.4	matched (3)
6	Bob	6	40	3.8	matched (3)
7	Pam	4	40	3.4	matched (3)
8	Pete	6	60	3.3	matched (3)
9	Phil	2	20	2.9	matched (3)
10	Alice	.	.	3.1	using only (2)

Command: *merge*

- If an observation in one data set does not match anything in the other data set, it will get missing values for all the variables in that data set.
- Stata gives you a report on screen on how successful you are at matching observations, it also creates a new variable, _*merge*, that tells you whether a given observation matched or not.
  - _*merge* $= 1$: the observation came from the master dataset but did not match anything in the using dataset.
  - _*merge* $= 2$: the observation came from the using dataset but did not match anything in the master dataset.
  - _*merge* $= 3$: successful!
- Note that you cannot carry out another merge until you drop or rename the _*merge* variable, so Stata can create a new one. And a more convenient way to store matching results is to create a new variable to replace _*merge*.

Command: *merge*

- Three commonly used merging patterns:
  * 1 : 1 merge: one sample in the master dataset will be matched with one sample in the using dataset. If we've specified that this is a 1:1 merge, the identifier variables must uniquely identify observations in both datasets.
  * 1 : m merge: one sample in the master dataset will be matched with many sample in the using dataset. (Consider the case where master dataset contains family-level data while using dataset contains individual-level data). If we've specified that this is a 1:m merge, the identifier variables must uniquely identify observations in the master dataset.
  * m : 1 merge: is essentially as same as 1 : m merge.

1:m merge

- Master database:

	famid	numkids	girls	boys
1	1	3	2	1
2	2	3	1	2
3	3	3	1	2

- Using database:

	famid	kidname	birth	age	wt	sex
1	1	Beth	1	9	60	f
2	1	Bob	2	6	40	m
3	1	Barb	3	3	20	f
4	2	Andy	1	8	80	m
5	2	Al	2	6	50	m
6	2	Ann	3	2	20	f
7	3	Pete	1	6	60	m
8	3	Pam	2	4	40	f
9	3	Phil	3	2	20	m

- What does the new dataset look like?

1:m merge

use kids_number.dta,clear
merge 1:m famid using original_data.dta

	famid	numkids	girls	boys	kidname	birth	age	wt	sex	_merge
1	1	3	2	1	Beth	1	9	60	f	matched (3)
2	2	3	1	2	Andy	1	8	80	m	matched (3)
3	3	3	1	2	Pete	1	6	60	m	matched (3)
4	1	3	2	1	Bob	2	6	40	m	matched (3)
5	1	3	2	1	Barb	3	3	20	f	matched (3)
6	2	3	1	2	Al	2	6	50	m	matched (3)
7	2	3	1	2	Ann	3	2	20	f	matched (3)
8	3	3	1	2	Pam	2	4	40	f	matched (3)
9	3	3	1	2	Phil	3	2	20	m	matched (3)

Command: *merge*

- Three commonly used merging patterns:
  - 1 : 1 merge: one sample in the master dataset will be matched with one sample in the using dataset. If we've specified that this is a 1:1 merge, the identifier variables must uniquely identify observations in both datasets.
  - 1 : m merge: one sample in the master dataset will be matched with many sample in the using dataset. (Consider the case where master dataset contains family-level data while using dataset contains individual-level data). If we've specified that this is a 1:m merge, the identifier variables must uniquely identify observations in the master dataset.
  - m : 1 merge: is essentially as same as 1 : m merge.

Question: Would I use 1:m merge or m:1 merge if my master dataset contains individuals' income levels for different years, and my using dataset contains unemployment rates for different years?

Command: *merge*

- Three commonly used merging patterns:
  - 1 : 1 merge: one sample in the master dataset will be matched with one sample in the using dataset. If we've specified that this is a 1:1 merge, the identifier variables must uniquely identify observations in both datasets.
  - 1 : m merge: one sample in the master dataset will be matched with many sample in the using dataset. (Consider the case where master dataset contains family-level data while using dataset contains individual-level data). If we've specified that this is a 1:m merge, the identifier variables must uniquely identify observations in the master dataset.
  - m : 1 merge: is essentially as same as 1 : m merge.
  - Never use m:m merge

Command: *merge*

The verification of *merge*:

- Think clearly about what the new dataset would look like before you start to merge.
- Then use commands like *count*, *sum*, *tab*, and the generated variable *_merge* to verify whether the combined dataset is consistent with your intention.
  * master: individual income
  * using: national-level unemployment rate
  - If I tab the unemployment rate, how many unique values would I get? How many times (Freq.) do each of these unique values appear?
- We cannot provide a checklist to ensure the merge result would definitely be correct. Thus, you need to learn the data very well and be very careful.

Combining two datasets

- Sometimes we need to combine two datasets into a new dataset.

append

* The *append* command creates a longer dataset.
* The same variables from the second dataset append after all the variables from the first dataset.
* If the same variable has different names in these two datasets, the variable name in one of the datasets should be changed by using the rename command so that the names match.

Command: *append*

- Basic syntax of *append*:

  append using filename_using_dataset

append example

	kidname	age	weight
1	Beth	9	60
2	Bob	6	40
3	Barb	3	20
4	Andy	8	80
5	Al	6	50
6	Ann	2	20
7	Pete	6	60
8	Pam	4	40
9	Phil	2	20

	kidname	age	weight
1	Beth	10	70
2	Bob	7	42
3	Barb	4	25
4	Andy	9	77
5	Al	7	53
6	Ann	3	24
7	Pete	7	66
8	Pam	5	32
9	Phil	3	30

```
use merge_base.dta,clear
append using merge_1.dta
```

append example

	kidname	age	weight
1	Beth	9	60
2	Bob	6	40
3	Barb	3	20
4	Andy	8	80
5	Al	6	50
6	Ann	2	20
7	Pete	6	60
8	Pam	4	40
9	Phil	2	20
10	Beth	10	70
11	Bob	7	42
12	Barb	4	25
13	Andy	9	77
14	Al	7	53
15	Ann	3	24
16	Pete	7	66
17	Pam	5	32
18	Phil	3	30

Figure 2: Append Result

append example

	kidname	age	weight
1	Beth	9	60
2	Bob	6	40
3	Barb	3	20
4	Andy	8	80
5	Al	6	50
6	Ann	2	20
7	Pete	6	60
8	Pam	4	40
9	Phil	2	20
10	Beth	10	70
11	Bob	7	42
12	Barb	4	25
13	Andy	9	77
14	Al	7	53
15	Ann	3	24
16	Pete	7	66
17	Pam	5	32
18	Phil	3	30

Figure 2: Append Result

Question: Is this a long-form dataset or a wide-form dataset?