Dataset Manipulation

LPO 9951 | Fall 2015

PURPOSE Learning to manipulate datasets is a key skill for statistical analysis. Today we'll work on four skills: subsetting data using preserve and restore commands, appending data, doing a simple one-to-one merge, and collapsing data.

Subsetting with `preserve` and `restore`

A feature and sometimes curse of Stata is that it can only hold one dataset in active memory at a time. As a feature, it helps you keep your work organized and, through numerous warning messages, tries to make sure you don't lose your work by accidentally forgetting to save or mindlessly overwritting your data. The feature feels more like a curse when you have multiple datasets that you would like to work with simultaneously or, as we will do below, split a single dataset into smaller parts.

To repeated subset a large dataset, there are two primary choices:
1. Reload the full dataset into memory after each subset and save
2. Use the `preserve` and `restore` commands

In the code below, notice how the `preserve` and `restore` commands bookend the `keep` command, which keeps only those observations that fulfill the `if` statement (in this case, the type of school). The steps are:
1. preserve dataset in memory
2. subset to keep only school type that we want
3. save new subset dataset
4. restore old dataset

```
. // elementary schools
. preserve

. keep if stype == 1
(1,773 observations deleted)

. save ${datadir}elem, replace
file ../data/elem.dta saved

. restore

. // high schools
. preserve

. keep if stype == 2
(5,439 observations deleted)

. save ${datadir}hs, replace
file ../data/hs.dta saved

. restore

. // middle schools (keep this one in memory so no preserve/restore needed)
. keep if stype == 3
(5,176 observations deleted)
```

```
. save ${datadir}middle, replace
file ../data/middle.dta saved
```

NB: Note the construction of the global macro to save the datasets in the proper subdirectory. Because we to paste the realization of the global (`../data/`) and the file name, we have to use the construction `${<global name>}` followed with no spaces by the name we want to give it. As you can see, `S{datadir}hs` becomes `../data/hs.dta`.

Appending Data

Appending data is done when we want to add additional *observations* to an existing dataset, using a dataset that has exactly the same variable names but different observations. Suppose you have data on high schools, middle schools, and elementary schools on a variety of performance indicators and you'd like to merge them together. The syntax uses, appropriately enough, the `append` command, which takes the format `append <new dataset>` (the command assumes the first dataset is the one in memory; remember that the middle school subset data are still in memory):

```
. append using ${datadir}elem
(label yr_rnd already defined)
(label awards already defined)
(label both already defined)
(label comp_imp already defined)
(label sch_wide already defined)
(label stype already defined)

. append using ${datadir}hs
(label stype already defined)
(label sch_wide already defined)
(label comp_imp already defined)
(label both already defined)
(label awards already defined)
(label yr_rnd already defined)
```

The `append` command will not copy over labels from the using dataset, so you'll need to make sure they're right in the master dataset. The most common error with an append command is to not have exactly matching variable names.

Merging Data

You can also use Stata's `merge` command to do an append operation in special cases. This happens when the merging variable doesn't have repeated *observations* in the two datasets, which in turn have exactly the same variable structure. Think of a Venn diagram where the circles contain exactly the same types of information, but don't overlap; in combining them, we've really just grown them into one bigger circle. One of the virtues of using `merge` when `append` will suffice is that you have access to more information about where the data came from once you're done.

```
. use ${datadir}elem, clear

. merge 1:1 snum using ${datadir}hs, gen(_merge_a)
(label yr_rnd already defined)
(label awards already defined)
```

```
(label both already defined)
(label comp_imp already defined)
(label sch_wide already defined)
(label stype already defined)

    Result                           # of obs.
    -----------------------------------------
    not matched                         5,176
        from master                     4,421  (_merge_a==1)
        from using                        755  (_merge_a==2)

    matched                                 0  (_merge_a==3)
    -----------------------------------------

. merge 1:1 snum using ${datadir}middle, gen(_merge_b)
(label stype already defined)
(label sch_wide already defined)
(label comp_imp already defined)
(label both already defined)
(label awards already defined)
(label yr_rnd already defined)

    Result                           # of obs.
    -----------------------------------------
    not matched                         6,194
        from master                     5,176  (_merge_b==1)
        from using                      1,018  (_merge_b==2)

    matched                                 0  (_merge_b==3)
    -----------------------------------------
```

Once you've completed the merge, you can take a look at the _merge_* variables that were generated to see where the data came from.

```
. tab _merge_a

           _merge_a |      Freq.     Percent        Cum.
------------------------+-----------------------------------
        master only (1) |      4,421       85.41       85.41
         using only (2) |        755       14.59      100.00
------------------------+-----------------------------------
                  Total |      5,176      100.00

. tab _merge_b

           _merge_b |      Freq.     Percent        Cum.
------------------------+-----------------------------------
        master only (1) |      5,176       83.56       83.56
         using only (2) |      1,018       16.44      100.00
------------------------+-----------------------------------
                  Total |      6,194      100.00
```

Quick Exercise

Create a dataset that has just middle and elementary schools. Do this using first the `append` command and then the `merge` command.

One-to-one merges

A one-to-one merge is when you have exactly the same *observations* but new variables to add to the dataset. Say you have *observations* with variables split across datasets, e.g., School 1 has variables A, B, and C in dataset 1 and variables X, Y, and Z in dataset two. As long as School 1 has a unique identifier—a name, an id number, etc—you can `merge` these two datasets together so that you have access to all of the school's variables for your analysis.

First, we need to subset our data again, only this time by splitting along columns (*variables*) rather than rows (*observations*):

```
. use $urldata, clear

. preserve

. keep snum api00 api99 ell meals         // variable set 1

. save ${datadir}api_1, replace
file ../data/api_1.dta saved

. restore

. keep snum full emer                     // variable set 2

. save ${datadir}api_2, replace
file ../data/api_2.dta saved
```

Now for the merge and view of merge stats:

```
. merge 1:1 snum using ${datadir}api_1

    Result                           # of obs.
    -----------------------------------------
    not matched                             0
    matched                             6,194  (_merge==3)
    -----------------------------------------

. // view merge stats
. tab _merge

                 _merge |      Freq.     Percent        Cum.
------------------------+-----------------------------------
            matched (3) |      6,194      100.00      100.00
------------------------+-----------------------------------
                  Total |      6,194      100.00
```

Quick Exercise

Create a dataset that has only mobility and percent tested. Next create another dataset that has only the year round and percent responding variables. Now merge these two datasets together using a one-to-one merge.

Collapsing data

Collapsing data refers to summarizing data across a type and creating a new dataset as a result. Say we want to create a county-level dataset from our school data, using the average figures for the schools across a set of characteristics. The command would look like this:

```
. // reload main dataset, since we didn't preserve it before
. use $urldata, clear

. // count of unique counties in dataset
. unique cnum
Number of unique values of cnum is  57
Number of records is  6194

. // mean of pcttest and mobility within countyr
. collapse (mean) pcttest mobility, by (cnum)

. // give count of number of observations (should be number of unique counties)
. count
  57
```

NB: Notice the check using the `unique` command. If the number didn't align or gave some unexpected value, we should recheck our data and code.

Quick Exercise

Create a district level dataset that contains district level averages for the following variables:
1. apioo
2. api99
3. ell
4. meals

Do the same thing using just district *medians.*

Init: 15 August 2015; Updated: 15 August 2015