Title

joinby — Form all pairwise combinations within groups

Description

joinby joins, within groups formed by varlist, observations of the dataset in memory with filename, a Stata-format dataset. By join we mean to form all pairwise combinations. filename is required to be sorted by varlist. If filename is specified without an extension, .dta is assumed.

If varlist is not specified, joinby takes as varlist the set of variables common to the dataset in memory and in filename.

Observations unique to one or the other dataset are ignored unless unmatched() specifies differently. Whether you load one dataset and join the other or vice versa makes no difference in the number of resulting observations.

If there are common variables between the two datasets, however, the combined dataset will contain the values from the master data for those observations. This behavior can be modified with the update and replace options.

Quick start

Form pairwise combinations of observations from mydata1.dta in memory with those from mydata2.dta using all common variables and drop unmatched observations

    joinby using mydata2

As above, but join on v1, v2, and v3

    joinby v1 v2 v3 using mydata2

As above, but include unmatched observations only from mydata2.dta and add _merge indicating whether the variable was in both datasets or only the using dataset

    joinby v1 v2 v3 using mydata2, unmatched(using)

As above, but include unmatched observations only from mydata1.dta

    joinby v1 v2 v3 using mydata2, unmatched(master)

As above, but name the variable indicating the source of the observation newv

    joinby v1 v2 v3 using mydata2, unmatched(master) _merge(newv)

Replace missing data in mydata1.dta with values from mydata2.dta

    joinby v1 v2 v3 using mydata2, update

Replace missing and conflicting data in mydata1.dta with values from mydata2.dta

    joinby v1 v2 v3 using mydata2, update replace
### Syntax

```
joinby [varlist] using filename [, options]
```

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>When observations match:</td>
<td></td>
</tr>
<tr>
<td>update</td>
<td>replace missing data in memory with values from <code>filename</code></td>
</tr>
<tr>
<td>replace</td>
<td>replace all data in memory with values from <code>filename</code></td>
</tr>
</tbody>
</table>

When observations do not match:

- `unmatched(none)` : ignore all; the default
- `unmatched(both)` : include from both datasets
- `unmatched(master)` : include from data in memory
- `unmatched(using)` : include from data in `filename`

- `_merge(varname)` : `varname` marks source of resulting observation; default is `_merge`
- `nolabel` : do not copy value-label definitions from `filename`

`varlist` may not contain `strL`.

### Options

- **update** varies the action that `joinby` takes when an observation is matched. By default, values from the master data are retained when the same variables are found in both datasets. If `update` is specified, however, the values from the using dataset are retained where the master dataset contains missing.

- `replace`, allowed with `update` only, specifies that nonmissing values in the master dataset be replaced with corresponding values from the using dataset. A nonmissing value, however, will never be replaced with a missing value.

- `unmatched(none | both | master | using)` specifies whether observations unique to one of the datasets are to be kept, with the variables from the other dataset set to missing. Valid values are
  - `none` : ignore all unmatched observations (default)
  - `both` : include unmatched observations from the master and using data
  - `master` : include unmatched observations from the master data
  - `using` : include unmatched observations from the using data

- `_merge(varname)` specifies the name of the variable that will mark the source of the resulting observation. The default name is `_merge(_merge)`. To preserve compatibility with earlier versions of `joinby`, `_merge` is generated only if `unmatched` is specified.

- `nolabel` prevents Stata from copying the value-label definitions from the dataset on disk into the dataset in memory. Even if you do not specify this option, label definitions from the disk dataset do not replace label definitions already in memory.
### Remarks and examples

The following, admittedly artificial, example illustrates `joinby`.

#### Example 1

We have two datasets: `child.dta` and `parent.dta`. Both contain a `family_id` variable, which identifies the people who belong to the same family.

```
. use https://www.stata-press.com/data/r16/child
   (Data on Children)
. describe
Contains data from https://www.stata-press.com/data/r16/child.dta
   obs: 5              Data on Children
   vars: 4             11 Dec 2018 21:08

   storage  display value
  variable name type format label
    family_id   int %8.0g   Family ID number
     child_id   byte %8.0g   Child ID number
          x1    byte %8.0g
          x2    int  %8.0g
Sorted by: family_id
. list
   +----------+-----------------+-------+
   | family-~d|     child_id  |  x1  |  x2  |
   |----------|----------------|-------|
   |   1025   |          3    |  11  | 320  |
   |   1025   |          1    |  12  | 300  |
   |   1025   |          4    |  10  | 275  |
   |   1026   |          2    |  13  | 280  |
   |   1027   |          5    |  15  | 210  |
   +----------+-----------------+-------+
```

```
. use https://www.stata-press.com/data/r16/parent
   (Data on Parents)
. describe
Contains data from https://www.stata-press.com/data/r16/parent.dta
   obs: 6              Data on Parents
   vars: 4             11 Dec 2018 03:06

   storage  display value
  variable name type format label
    family_id  int  %8.0g   Family ID number
     parent_id float %9.0g   Parent ID number
          x1    float %9.0g
          x3    float %9.0g
Sorted by: 
```
We want to join the information for the parents and their children. The data on parents are in memory, and the data on children are posted at https://www.stata-press.com. child.dta has been sorted by family_id, but parent.dta has not, so first we sort the parent data on family_id:

```
. sort family_id
. joinby family_id using https://www.stata-press.com/data/r16/child
. describe
```

Contains data
```
obs: 8
Data on Parents
vars: 6
```

<table>
<thead>
<tr>
<th>variable name</th>
<th>storage</th>
<th>display</th>
<th>value</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>family_id</td>
<td>int</td>
<td>%8.0g</td>
<td>label</td>
<td>Family ID number</td>
</tr>
<tr>
<td>parent_id</td>
<td>float</td>
<td>%9.0g</td>
<td>Parent ID number</td>
<td></td>
</tr>
<tr>
<td>x1</td>
<td>float</td>
<td>%9.0g</td>
<td>label</td>
<td></td>
</tr>
<tr>
<td>x3</td>
<td>float</td>
<td>%9.0g</td>
<td>label</td>
<td></td>
</tr>
<tr>
<td>child_id</td>
<td>byte</td>
<td>%8.0g</td>
<td>Child ID number</td>
<td></td>
</tr>
<tr>
<td>x2</td>
<td>int</td>
<td>%8.0g</td>
<td>label</td>
<td></td>
</tr>
</tbody>
</table>

Sorted by: family_id

Note: Dataset has changed since last saved.

```
. list, sepby(family_id) abbrev(12)
```

```
<table>
<thead>
<tr>
<th>family_id</th>
<th>parent_id</th>
<th>x1</th>
<th>x3</th>
<th>child_id</th>
<th>x2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1030</td>
<td>10</td>
<td>39</td>
<td>600</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1025</td>
<td>11</td>
<td>20</td>
<td>643</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1025</td>
<td>12</td>
<td>27</td>
<td>721</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1026</td>
<td>13</td>
<td>30</td>
<td>760</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1026</td>
<td>14</td>
<td>26</td>
<td>668</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1030</td>
<td>15</td>
<td>32</td>
<td>684</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1027</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1030</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

1. family_id of 1027, which appears only in child.dta, and family_id of 1030, which appears only in parent.dta, are not in the combined dataset. Observations for which the matching variables are not in both datasets are omitted.
2. The $x_1$ variable is in both datasets. Values for this variable in the joined dataset are the values from `parent.dta`—the dataset in memory when we issued the `joinby` command. If we had `child.dta` in memory and `parent.dta` on disk when we requested `joinby`, the values for $x_1$ would have been those from `child.dta`. Values from the dataset in memory take precedence over the dataset on disk.

Acknowledgment

`joinby` was written by Jeroen Weesie of the Department of Sociology at Utrecht University, The Netherlands.

Reference

Baum, C. F. 2016. *An Introduction to Stata Programming*. 2nd ed. College Station, TX: Stata Press.

Also see

[D] `append` — Append datasets  
[D] `cross` — Form every pairwise combination of two datasets  
[D] `fillin` — Rectangularize dataset  
[D] `merge` — Merge datasets  
[D] `save` — Save Stata dataset  
[U] 23 Combining datasets