Title

intro — Introduction to data-management reference manual

Description

This entry describes this manual and what has changed since Stata 10. See the next entry, [D] **data management**, for an introduction to Stata's data-management capabilities.

Remarks

This manual documents most of Stata's data-management features and is referred to as the [D] manual. Some specialized data-management features are documented in such subject-specific reference manuals as [TS] *Time-Series Reference Manual*, [ST] *Survival Analysis and Epidemiological Tables Reference Manual*, and [XT] *Longitudinal-Data/Panel-Data Reference Manual*.

Following this entry, [D] **data management** provides an overview of data management in Stata and of Stata's data-management commands. The other parts of this manual are arranged alphabetically. If you are new to Stata's data-management features, we recommend that you read the following first:

[D] **data management** — Introduction to data-management commands
[U] **12 Data**
[U] **13 Functions and expressions**
[U] **11.5 by varlist: construct**
[U] **21 Inputting data**
[U] **22 Combining datasets**
[U] **23 Working with strings**
[U] **25 Working with categorical data and factor variables**
[U] **24 Working with dates and times**
[U] **16 Do-files**

You can see that most of the suggested reading is in [U]. That is because [U] provides overviews of most Stata features, whereas this is a reference manual and provides details on the usage of specific commands. You will get an overview of features for combining data from [U] **22 Combining datasets**, but the details of performing a match-merge (merging the records of two files by matching the records on a common variable) will be found here, in [D] **merge**.

Stata is continually being updated, and Stata users are always writing new commands. To ensure that you have the latest features, you should install the most recent official update; see [R] **update**.

What's new

This section is intended for previous Stata users. If you are new to Stata, you may as well skip it.

1. Stata has an all-new data editor!

   The Data Editor now really is a live view onto the data. That means you can leave the Data Editor up while you run a Stata command—including a data-management command—and when it finishes, the Editor will update its view.

Inside the Data Editor, you can drop variables or observations, generate new variables, replace the contents of existing variables, and even sort observations. In fact, you can run any data-management command with the Data menu or by typing in the Command window. And no matter how you modify your data, all data editing is translated to real commands that appear in the Review window. Well, no matter how you modify your data, assuming you are not pasting from the clipboard.

The Data Editor can hide and, more importantly, unhide variables or observations. Observations can be selected by expression.

The Editor has live filtering. If you filter observations with an expression and change one or more of the variables in the expression, no matter how, the view updates immediately.

You can take one or more snapshots of your data as you are working in the Data Editor and then restore your data from a snapshot should you make a mistake while editing.

The Editor has improved keyboard and mouse navigation. Editing can be performed in place, right where the cursor is, and you can jump to a cell by typing a portion of the variable's name and observation number. All works whether you are editing or doing something else.

You can easily input dates and times!

Finally, you can shift from edit to browse modes, and back again.

Select **Data** > **Data Editor (Edit)**, or type `edit`. See [D] **edit** and [GS] **6 Using the Data Editor** (GSM, GSU, or GSW).

2. Try the new variable management features; select **Data** > **Variables Manager** or type `varmanage`. You can then select a variable or multiple variables. You can select variables the normal way, or you can type in the top-left box and the Variables Manager will filter the list. You can change the storage type, variable name, and format, and you can add or edit the value labels and even the notes. See [D] **varmanage** and [GS] **7 Using the Variables Manager** (GSM, GSU, or GSW).

3. The Do-file Editor is all new under Windows. It provides syntax highlighting, code folding, line bookmarking, and line numbering. Syntax highlighting means commands and keywords, functions, macros, strings, and comments are shown in different colors. Code folding means code blocks bound by braces can be collapsed (or expanded). Line bookmarking means that you can attach a bookmark to a line for quick access later. Your do-file can have multiple bookmarks. File size is limited only by the availability of memory. See [R] **doedit** and [GS] **13 Using the Do-file Editor—automating Stata** (GSM, GSU, or GSW).

4. Existing command `merge` has all new syntax. It is easier to use, easier to read, and makes it less likely that you will obtain an unintended result. Merges are classified as `1:1`, `1:m`, `m:1`, and `m:m`. When you type `merge 1:1 subjid`, you are saying that you expect the observations to match one-to-one, what was called `uniqmaster` and `uniqusing` in the old syntax. Classification `1:m` specifies a 1-to-many merge; `m:1`, a many-to-1 merge; and `m:m`, a many-to-many merge. New options `assert()` and `keep()` allow you to specify what you expect and what you want to keep, so `merge 1:1 subjid using` *filename*`, assert(match)` means that you expect all the observations in both datasets to match each other. Sorting of both the master and using datasets is now automatic.

   The new `merge` does not support merging multiple files in one step. Merge the first two datasets, then merge that with the next dataset, and so on.

   See [D] **merge**. The old `merge` syntax continues to work.

5. Existing command `append` has several new features. It will work even if there is no data in memory. Multiple files can be appended in one step. New option `generate(`*newvar*`)` creates a variable indicating the source of the observations, numbered 0, 1, ... `append` now aborts with

error if you attempt to match a string variable with a numeric unless option `force` is specified. See [D] **append**. Old behavior is preserved under version control.

6. Existing command `order` is really all new and does what the previous commands `order`, `move`, and `aorder` did. See [D] **order**. Old commands `aorder` and `move` continue to work but are no longer documented.

7. New commands `zipfile` and `unzipfile` compress and uncompress files (and directories) in zip archive format. See [D] **zipfile**.

8. New command `changeeol` converts text from one operating system's end-of-line format to another. Stata does not care about end-of-line format, but some editors and other programs do. See [D] **changeeol**.

9. New command `snapshot` saves to disk and restores from disk copies of the data in memory. `snapshot`'s main purpose is to allow the Data Editor to save and restore data snapshots during an interactive editing session. See [D] **snapshot**.

10. Existing command `notes` has new options `search`, `replace`, and `renumber`. See [D] **notes**.

11. Concerning value labels:

    a. Existing command `label define` has new option `replace` so that you do not have to drop the value label first.

    b. New command `label copy` copies value labels.

    c. Existing command `label values` now allows a varlist, so you can label (or unlabel) a group of variables at the same time.

    See [D] **label**.

12. Existing command `expand` has new option `generate(`*newvar*`)` that makes it easier to distinguish original from duplicated observations. See [D] **expand**.

13. Concerning `egen`:

    a. New function `rowmedian(`*varlist*`)` returns, observation by observation, the median of the values in *varlist*.

    b. New function `rowpctile(`*varlist*`), p(`*#*`)` returns, observation by observation, the *#*th row percentile of the values within *varlist*.

    c. Existing function `mode(`*varname*`)` with option `missing` treats missing values as a category. When version is set to 10 or less, `missing` does not treat missing as a category.

    d. Existing function `total(`*exp*`)` and `rowtotal(`*varlist*`)` have new option `missing`. If all values of *exp* or *varlist* for an observation are missing, then that observation in *newvar* will be set to missing.

    See [D] **egen**.

14. Existing command `copy` now allows copying a file to a directory without having to type the filename twice; see [D] **copy**.

15. Existing command `clear` now allows `clear matrix` to clear all Stata matrices (not Mata matrices) from memory; see [D] **clear**.

16. Existing command `outfile` now exports date variables as strings rather than their underlying numeric value. Under version control, old behavior is restored. See [D] **outfile**.

17. Existing command `reshape` now preserves variable and value labels when converting from long to wide and restores variable and value labels when converting from wide to long. Thus the value and variable labels for the `i` variable, which exists in long form and not in wide form, are restored

when converting back from wide to long. The value labels of the `xij` variables are similarly restored. Prior behavior is preserved when version is 10 or earlier. See [D] **reshape**.

18. Existing command `collapse` now allows new statistics `semean`, `sebinomial`, and `sepoisson` for obtaining the standard error of the mean. See [D] **collapse**.

19. Existing command `destring` allows new option `dpcomma` to convert to numeric form string representation of numbers using commas as the decimal point. See [D] **destring**.

20. Concerning existing command `odbc`:

    a. `odbc insert` now uses parameterized inserts, which are faster.

    b. The dialogs for `odbc load` and `odbc insert` can now store a data-source user ID and password for a Stata session.

    c. `odbc query` has new options `verbose` and `schema`. `verbose` lists any data source alias, nickname, typed table, typed view, and view along with tables so that data from these table types can be loaded. `schema` lists schema names with the table names if the data source returns schema information.

    d. `odbc insert` has a new dialog.

    e. Existing option `dsn()` now allows the data source to be up to 499 characters.

    f. `odbc` now reports driver errors directly. Previously, `odbc` would issue the error "ODBC error; type -set debug on- and rerun command to see extended error information" when an ODBC driver issued an error.

    g. `odbc`, with `set debug on`, for security reasons no longer displays the data source name, user ID, and password used for connecting to your data source.

    See [D] **odbc**.

21. New function `strtoname()` converts a general string to a string meeting Stata's naming conventions. Also, existing functions `lower()`, `ltrim()`, `proper()`, `reverse()`, `rtrim()`, and `upper()`, now have synonyms `strlower()`, `strltrim()`, ..., `strupper()`. Both sets of names work equally well. See [D] **functions**.

22. New function `soundex()` returns the soundex code for a name, consisting of a letter followed by three numbers. New function `soundex_nara()` returns the U.S. Census soundex for a name, also consisting of a letter followed by three numbers, but produced by a different algorithm. See [D] **functions**.

23. New functions `sinh()`, `cosh()`, `asinh()`, and `acosh()` join existing functions `tanh()` and `atanh()` to provide the hyperbolic functions. See [D] **functions**.

24. New functions `binomialp()`; `hypergeometric()` and `hypergeometricp()`; `nbinomial()`, `nbinomialp()`, and `nbinomialtail()`; and `poisson()`, `poissonp()`, and `poissontail()` provide distribution and probability mass for the binomial, hypergeometric, negative binomial, and Poisson distributions. See [D] **functions**.

25. New functions `invnbinomial()` and `invnbinomialtail()`, and `invpoisson()` and `invpoissontail()` provide inverses for the negative binomial and Poisson distributions. See [D] **functions**.

26. Algorithms for the existing functions `normal()` and `lnnormal()` have been improved to operate in 60% and 75% of the time, respectively, while giving equivalent double-precision results.

27. New functions `rbeta()`, `rbinomial()`, `rchi2()`, `rgamma()`, `rhypergeometric()`, `rnbinomial()`, `rnormal()`, `rpoisson()`, and `rt()` produce random variates for the beta, binomial, chi-squared, gamma, hypergeometric, negative binomial, normal, Poisson, and Student's t distributions.

Old function `uniform()`has been renamed to `runiform()`, but `uniform()` continues to work.

Thus all random-variate functions start with `r`.

See [D] **functions**.

28. Existing command `drawnorm` now uses new function `rnormal()` to generate random variates. When version is set to 10 or earlier, `drawnorm` reverts to using `invnormal(uniform())`. See [D] **functions**.
29. Existing command `describe` now respects the width of the Results window when formatting output; see [D] **describe**.
30. Existing command `renpfix` now returns the list of variables changed in `r(varlist)`; see [D] **rename**.
31. Previously existing command `impute` still works but is now undocumented. It is replaced by the new multiple-imputation command `mi`. See the *Multiple-Imputation Reference Manual*.

For a complete list of all the new features in Stata 11, see [U] **1.3 What's new**.

Also see

[U] **1.3 What's new**

[R] **intro** — Introduction to base reference manual