

Title

intro — Introduction to data-management reference manual

Description

This entry describes this manual and what has changed since Stata 8. 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] *Stata Time-Series Reference Manual*, [ST] *Stata Survival Analysis and Epidemiological Tables Reference Manual*, and [XT] *Stata Longitudinal/Panel Data Reference Manual*.

Following this entry, [D] **data management** provides an overview of data management in Stata and 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 Dealing with strings**

[U] **25 Dealing with categorical variables**

[U] **24 Dealing with dates**

[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. This manual. It documents most of the data-management commands, many of which were previously spread through the pages of [R]. No division of data-management and analysis commands will meet everyone's expectations, but we believe that learning and referencing Stata's data-management capabilities will be easier with this new arrangement.

Note that some data-management commands are intended for special types of data, and those commands remain documented in the subject-specific manuals; see, for example, [ST] **sts generate**, [ST] **snapspan**, [XT] **xtdes**, and [TS] **tsfill**.

2. You can now permanently set the default data type to either `float` (the factory default) or `double`. The default type setting determines the type for almost all variables created in Stata—those created by `generate`, `infile`, `insheet`, etc. Enter the command `set type double`, permanently to change the default type to `double`. Note that this will effectively double the size of most datasets but will allow you to process numbers with more than 7 digits of accuracy, such as Social Security or driver's license numbers, without explicitly declaring the variable to be `double`.
3. New commands `xmlsave` and `xmluse` save and restore datasets in XML (Extended Markup Language) format. Data may be saved and used in either Stata `dta` XML format or Microsoft Excel's SpreadsheetML format. See [D] **xmlsave**.
4. New commands `fdasave`, `fdause`, and `fdadescribe` save, use, and describe data files in SAS XPORT format, the format used by the U.S. Food and Drug Administration (FDA) for new drug and device applications (NDAs). These commands are designed to assist people making submissions to the FDA, but the commands are general enough for use in transferring data between SAS and Stata. See [D] **fdasave**.
5. `label` has the new subcommand `language` that lets you create datasets containing multiple sets of data, variable, and value labels. A dataset might contain one set in English, another in German, and a third in Spanish. Another dataset might contain one set of short labels and another of long labels. Either way, you can switch between the label sets as desired. Up to 100 sets of labels are allowed.

`codebook`, `labelbook`, and `uselabel` support multiple-language variable and value labels. See [D] **label language**.
6. Value labels may now be up to 32,000 characters long.
7. Datasets from the examples in the Stata manuals can now be browsed by command name, described, and used. Type `help dta_contents`, or select **File > Example Datasets...** from the Stata menu.
8. `statsby` is now a prefix command, see [U] **11.1.10 Prefix commands**, and its new syntax is

$$\text{statsby } [exp_list] [, options] : command$$

The old syntax of `statsby` continues to work but has none of the new features and enhancements. Enhancements to `statsby` are

- a. Rather than always requiring a list of expressions for the results to collect, `statsby` now collects a default set of statistics appropriate to the command being run. All estimated parameters are collected under their parameter names for estimation commands, and all scalar results saved in `r()` are collected under their `r(name)` names for nonestimation commands. You can still specify explicit lists of results to collect.
- b. Expressions to be computed and saved can now be grouped under prefixes to be given to the names of the collected statistics using the new `equation:` directive; see `help exp_list`.
- c. String variables are now allowed in the `by()` option, and the new `missing` suboption includes missing values in the by-groups.

- d. Weights can now be specified on the command being run.
- e. The new `force` option forces `statsby` to work with survey estimators. By default, this is prevented because the method `statsby` uses to select subsamples will generally not produce appropriate standard-error estimates with survey data (the `subpop` option must be used with survey data).
- f. The `saving()` option now bundles its modifiers into suboptions; i.e.,

```
saving(filename [ , double every(#) replace ])
```
- g. Dots showing the progress of computations are now shown by default and new option `nodots` suppresses the display.
- h. The new option `nolegend` suppresses the new table reporting on what `statsby` is running.
9. New command `filefilter` copies an input file to an output file while converting a specified ASCII text string or binary pattern to another pattern. This is a way to edit large datasets and make them suitable for use with `infile` or `infix`; see [D] **filefilter**.
10. New command `expandcl` replicates clusters of unique observations, much like an `expand` replicates observations; see [D] **expandcl**.
11. `insheet` now creates floating-point variables as `double` if the default type is set to `double`. `insheet` has the new option `nodouble` that forces `insheet` to store floating-point variables as floats, even if `set type` has been set to `double`.
12. New command `tostring` converts numeric variables to string variables; see [D] **destring**.
13. `codebook` now allows `if exp` and `in range` qualifiers; see [D] **codebook**.
14. New command `rmdir` removes an empty directory (folder); see [D] **rmdir**.
15. New command `clonevar` makes an identical copy of an existing variable; see [D] **clonevar**.
16. `icd9` and `icd9p` have been updated to use the V21 codes; V19, V18, and V16 codes were previously used. V16, V18, V19, and V21 codes have been merged so that `icd9` and `icd9p` work equally well with old and new datasets. See [D] **icd9** for a description of `icd9` and `icd9p`; type `icd9 query` and `icd9p query` for a complete description of the changes to the codes used.
17. `encode` has the new option `noextend` that modifies the existing `label()` option and restricts encoding to labels in the specified value label; see [D] **encode**.

(Continued on next page)

18. The following egen functions have been renamed:

old name	new name
any()	anyvalue()
eqany()	anymatch()
neqany()	anycount()
rfirst()	rowfirst()
rlast()	rowlast()
rmean()	rowmean()
rmin()	rowmin()
rmiss()	rowmiss()
robs()	rownonmiss()
rsd()	rowsd()
rsum()	rowtotal()
sum()	total()

The old names continue to work but are not documented.

19. The `odbc` command for accessing Open DataBase Connectivity (ODBC) data sources has the following enhancements:

- a. ODBC is now supported under Mac OS X and Linux systems that use the ODBC Driver Manager. For more information on configuring ODBC for Mac and Linux, see the FAQ at <http://www.stata.com/support/faqs/data/odbcmu.html>.
- b. `odbc` has new subcommands `odbc insert` and `odbc exec` for writing data to an ODBC data source. Positioned updates can be performed using the `odbc exec` command.
- c. `odbc` has the new subcommand `sqlfile` for batch processing SQL instructions.
- d. `odbc load` has the new option `sqlshow` for debugging SQL communication with ODBC drivers.
- e. `odbc load` has two new options, `allstring` and `datestring`, which import either all data or just dates as strings.

See [D] **odbc** for more information.

20. `merge` has the following new features.

- a. `merge` now accepts multiple files.
- b. The new option `nosummary` suppresses creating variables that summarize how records were merged.
- c. The new option `sort` sorts the master and using datasets if necessary.
- d. Existing options `unique`, `uniquemaster`, and `uniquing` now require you to specify match variables.
- e. Warning messages are now given when match variables do not uniquely identify observations in either the master or the using datasets.

See [D] **merge**.

21. `merge` and `append` now incorporate all notes from the using dataset that do not appear in the master dataset. A new option, `nonotes`, ignores the notes in the using dataset, which was the previous behavior. See [D] **merge** and [D] **append**.

22. The Data Editor, which can be invoked using the tool bar, the menus, or the `edit` command, has new features.
- You can now define, modify, and attach value labels to variables.
 - Strings and value labels are now color-coded differently so that they can be easily distinguished.
- See [D] **edit** for details.
23. `contract` has five new options—`cfreq()`, `percent()`, `cpercent()`, `float`, and `format()`—to create frequency and percentage variables; see [D] **contract**.
24. `corr2data` and `drawnorm` now support triangular specification of the correlation or covariance matrix. See the discussion of the new `cstorage()` option in [D] **corr2data** and [D] **drawnorm**.
25. `separate` now has the new option `shortlabel` to specify that shorter variable labels be created; see [D] **separate**.
26. `outfile` has the new option `missing` that preserves both standard and extended missing values when the `comma` option is also specified; see [D] **outfile**.
27. The following functions have been renamed:

old name	new name
<code>binorm()</code>	<code>binormal()</code>
<code>index()</code>	<code>strpos()</code>
<code>invnorm()</code>	<code>invnormal()</code>
<code>issym()</code>	<code>issymmetric()</code>
<code>lnfact()</code>	<code>lnfactorial()</code>
<code>match()</code>	<code>strmatch()</code>
<code>normden()</code>	<code>normalden()</code>
<code>norm()</code>	<code>normal()</code>
<code>syminv()</code>	<code>invsym()</code>

The old names are no longer documented but continue to work. In most cases, the renamings were done to make function names consistent with those in Mata; see *Mata Reference Manual*.

New functions have been added: `trunc()`, which is a synonym for `int()`; `strlen()`, which is a synonym for `length()`; and `stofreal()`, which is a synonym for `string()`. See [D] **functions**. Again, the synonyms were added to make function names consistent with those in Mata; see *Mata Reference Manual*.

Also See

Complementary: [U] 1.3 What's new

Background: [R] intro