

Title

intro — Introduction to data-management reference manual

Description

This entry describes this manual and what has changed since Stata 9. 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 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 10 has new date/time variables, so you can now record values like 14jun2007 09:42:41.106 in one variable. They are called %tc and %tC variables. The first is unadjusted for leap seconds; the second is adjusted.

What used to be called “daily variables” are now called `%td` variables. This is just a jargon change; daily (`%td`) variables continue to work as they did before—0 means 01jan1960, 1 means 02jan1960, and so on.

`%tc` and `%tC` variables work similarly: 0 means 01jan1960 00:00:00. Here, however, 1 means 01jan1960 00:00:00.001, 1000 means 01jan1960 00:00:01.000, and 02jan1960 08:00:00 is 115,200,000. The underlying values are big—so it is important you store them as doubles—but the `%tc` and `%tC` formats make the values readable, just as the `%td` format makes daily (`%td`) values readable.

There are many new functions to go along with this new value type. `clock()`, for instance, converts strings such as “02jan1960 08:00:00” (or even “8:00 a.m., 1/2/1960”) to their numeric equivalents. `dofc()` converts a `%tc` value (such as 115,200,000, meaning 02jan1960 08:00:00) to its `%td` equivalent (namely, 1, meaning 02jan1960). `cofd()` does the reverse (the result would be 86,400,000, meaning 02jan1960 00:00:00).

See [D] **dates and times**.

2. The previously existing `date()` function, which converts strings to `%td` values, is now smarter. In addition to being able to convert strings such as “21aug2005”, “August 21, 2005”, it can convert “082105”, “08212005”, “210805”, and “21082005”. See [D] **dates and times**.
3. New command `datasignature` allows you to sign datasets and later use that signature to determine whether the data have changed. An early version of the command was made available during the Stata 9 release. That command is now called `_datasignature` and was used as the building block for the new, improved `datasignature`. See [D] **datasignature** and [P] **_datasignature**.
4. Existing command `clear` now clears data and value labels only. Type `clear all` to clear everything. This change will bite you the first few times you type `clear` expecting it to clear all. The problem was that new users were surprised when `clear` by itself cleared everything, whereas use `filename`, `clear` loaded new data and value labels but left everything else in place. The new users were right.

`clear` now has the following subcommands:

- a. `clear all` clears everything from memory.
- b. `clear ado` clears automatically loaded ado-file programs.
- d. `clear programs` clears all programs, automatically loaded or not.
- c. `clear results` clears saved results.
- d. `clear mata` clears Mata functions and objects from memory.

See [D] **clear**.

5. Stata for Unix now supports `unixODBC` [*sic*], making it easier to connect to databases such as Oracle, MySQL, and PostgreSQL; see [D] **odbc**.
6. Existing command `describe` now allows option `varlist` that was previously allowed only by `describe using`. Existing command `describe using filename` now allows option `simple` that was previously allowed only by `describe`. Option `varlist` saves the variable names in `r(varlist)`, and option `simple` displays the variable names in a compact format. See [D] **describe**.
7. Existing command `collapse` now supports four additional `stats`: `first`, the first value; `last`, the last value; `firstnm`, the first nonmissing value; and `lastnm`, the last nonmissing value. See [D] **collapse**.

8. Existing command `cf` (compare files) now provides a detailed listing of observations that differ when the `verbose` option is specified. Setting version to less than 10.0 restores the earlier behavior. See [D] **cf**.
9. Existing command `codebook` has new option `compact` that produces more compact output. See [D] **codebook**.
10. Existing command `insheet` has new option `case` that preserves the case of variable names when importing data; see [D] **insheet**.
11. Existing command `outsheet` has new option `delimiter()` that specifies an alternative delimiter; see [D] **outsheet**.
12. Existing commands `infile` and `infix` can now read up to 524,275 characters per line; the previous limit was 32,765. See [D] **infile** and [D] **infix (fixed format)**.
13. Existing commands `icd9` and `icd9p` have now been updated to use the V24 codes; see [D] **icd9**.
14. New function `itrim()` returns the string with consecutive, internal spaces collapsed to one space; see *String functions* in [D] **functions**.
15. New functions `lnnormal()` and `lnnormalden()` provide the natural logarithm of the cumulative standard normal distribution and of the standard normal density; see *Probability distributions and density functions* in [D] **functions**.
16. New functions for calculating cumulative densities are now available:

<code>binomial(<i>n</i>, <i>k</i>, <i>p</i>)</code>	lower tail of the binomial distribution
<code>ibetatail(<i>a</i>, <i>b</i>, <i>x</i>)</code>	reverse (upper tail) of the cumulative beta distribution
<code>gammaptail(<i>a</i>, <i>x</i>)</code>	reverse (upper tail) of the cumulative gamma distribution
<code>invgammaptail(<i>a</i>, <i>p</i>)</code>	inverse reverse of the cumulative gamma distribution
<code>invibetatail(<i>a</i>, <i>b</i>, <i>p</i>)</code>	inverse reverse of the cumulative beta distribution
<code>invbinomialtail(<i>n</i>, <i>k</i>, <i>p</i>)</code>	inverse of right cumulative binomial

See *Probability distributions and density functions* in [D] **functions**.

17. Existing function `Binomial(n, k, p)` has been renamed `binomialtail(n, k, p)`, thus making its name consistent with the naming convention for probability functions. The accuracy of the function has also been improved for very large values of *n*. At the other end of the number line, the function now returns the appropriate 0 or 1 value when *n* = 0, rather than returning missing. `Binomial()` continues to work as a synonym for `binomialtail()`.
18. The behavior and accuracy of the following probability functions have been improved:
 - a. `F(n1, n2, f)` and `Ftail(n1, n2, f)` are more accurate for small values of *n*₁ and large values of *n*₂. Also, `F()` is more accurate for large *f* where *n*₁ and *n*₂ are less than 1.
 - b. `gammap(a, x)` is more accurate when *a* is large and *x* is near *a*.
 - c. `ibeta(a, b, x)` now is more accurate when *x* is near *a*/(*a* + *b*) and *a* or *b* is large.
 - d. `invbinomial(n, k, p)`, `invchi2(n, p)`, `invchi2tail(n, p)`, `invF(n1, n2, p)`, and `invgammap(a, p)` are more accurate for small values of *p* or for returned values close to zero.
 - e. `invFtail(n1, n2, p)` and `invibeta(a, b, p)` are more accurate for small values of *p* or for returned values close to zero.
 - f. `invttail(n, p)` is more accurate for small values of *p* or for returned values close to zero.
 - g. `ttail(n, t)` is more accurate for exceedingly large values of *n*.

19. Existing function `invbinomial(n, k, p)` now returns the probability of a success on one trial such that the probability of observing *k* or *fewer* successes in *n* trials is *p*. The previous behavior of `invbinomial()` is restored under version control.
20. New function `fmtwidth()` returns the display width of a *%fmt* string; see *Programming functions* in [D] **functions**.
21. The maximum length of a *%fmt* has increased from 12 to 48 characters; see [D] **format**. (This change was necessitated by the new date/time variables.)
22. Existing commands `corr2data` and `drawnorm` now allow singular correlation (or covariance) structures. New option `forcepsd` modifies a matrix to be positive semidefinite and thus to be a proper covariance matrix. See [D] **corr2data** and [D] **drawnorm**.
23. Existing command `hexdump`, `analyze` now saves the number of `\r\n` characters in `r(Windows)` rather than in `r(DOS)`. `r(DOS)` is still set when version is less than 10. See [D] **hexdump**.

For a complete list of all the new features in Stata 10, see [U] **1.3 What's new**.

Also See

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

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