

<b>table</b> — Table of frequencies, summaries, and command results
---

<a href="#">Description</a>	<a href="#">Quick start</a>	<a href="#">Menu</a>	<a href="#">Syntax</a>
<a href="#">Options</a>	<a href="#">Remarks and examples</a>	<a href="#">Stored results</a>	<a href="#">Methods and formulas</a>
<a href="#">Also see</a>			

## Description

`table` is a flexible command for creating tables of many types—tabulations, tables of summary statistics, tables of regression results, and more. `table` can calculate summary statistics to display in the table. `table` can also include results from other Stata commands.

## Quick start

Two-way tabulation of `a1` and `a2`

```
table a1 a2
```

Table of means for `v1` and `v2` across the levels of `a1`

```
table a1, statistic(mean v1 v2)
```

Two-way table with `a1` defining rows and `a2` defining columns, with frequencies and pairwise correlation coefficients between `v3` and `v4` computed for every cell

```
table a1 a2, command(pwcorr v3 v4)
```

Table of regression coefficients with means of the covariates; rows correspond to covariates and columns correspond to the statistics

```
table (colname) (statcmd result),
    command(regress y x1 x2)
    statistic(mean x1 x2)
```

Same as above, and include standard deviations for the covariates

```
table (colname) (statcmd result),
    command(regress y x1 x2)
    statistic(mean x1 x2)
    statistic(sd x1 x2)
```

## Menu

Statistics > Summaries, tables, and tests > Tables of frequencies, summaries, and command results

## Syntax

*Basic syntax for a one-way table*

```
table rowvar  
table () colvar
```

*Basic syntax for a two-way table*

```
table rowvar colvar
```

*Basic syntax for an n-way table*

```
table rowvars colvar  
table rowvar (colvars)  
table (rowvars) (colvars)
```

*Basic syntax for multiple n-way tables*

```
table (rowvars) (colvars) (tabvars)
```

*Full syntax*

```
table (rowspec) (colspec) [(tabspec)] [if] [in] [weight] [, options]
```

*rowspec*, *colspec*, and *tabspec* may be empty or may include variable names or any of the following keywords:

<i>keyword</i>	Description
<b>result</b>	requested statistics
<b>stars</b>	stars denoting statistical significance
<b>var</b>	variables from <code>statistic()</code> option
<b>across</b>	index <code>across()</code> specifications
<b>colname</b>	column names for matrix statistics
<b>rowname</b>	row names for matrix statistics
<b>coleq</b>	column equation names for matrix statistics
<b>roweq</b>	row equation names for matrix statistics
<b>command</b>	index option <code>command()</code>
<b>statcmd</b>	index options <code>statistic()</code> and <code>command()</code>

<i>options</i>	Description
Main	
<code>totals(<i>totals</i>)</code>	report only the specified totals
<code>nototals</code>	suppress the marginal totals
Statistics	
<code><u>statistic</u>(<i>statspec</i>)</code>	statistic to be reported; default is <code>statistic(frequency)</code> when no weights are specified and <code>statistic(sumw)</code> otherwise
Commands	
<code>command(<i>cmdsproc</i>)</code>	collect results from the specified Stata command
Formats	
<code>nformat(<i>%fmt</i> [<i>results</i>][, <i>basestyle</i>])</code>	specify numeric format
<code>sformat(<i>sfmt</i> [<i>results</i>])</code>	specify string format
<code>cidelimiter(<i>char</i>)</code>	use character as delimiter for confidence interval limits
<code>cridelimiter(<i>char</i>)</code>	use character as delimiter for credible interval limits
Stars	
<code>stars(<i>starspec</i>)</code>	add stars to denote statistical significance
Options	
<code>listwise</code>	use listwise deletion to handle missing values
<code>missing</code>	treat numeric missing values like other values
<code>showcounts</code>	show sample size for all variables in <code>statistic()</code> option
<code>zerocounts</code>	report 0 for empty cell counts
<code>name(<i>cname</i>)</code>	collect results into a collection named <i>cname</i>
<code>append</code>	append results to an existing collection
<code>replace</code>	replace results of an existing collection
<code>label(<i>filename</i>)</code>	specify the collection labels
<code>style(<i>filename</i> [, <i>override</i>])</code>	specify the collection style
<code>markvar(<i>newvar</i>)</code>	create <i>newvar</i> that identifies observations used in the tabulation
<code>noisily</code>	display output from each command

`fweights`, `awweights`, `iwweights`, and `pweights` are allowed; see [U] 11.1.6 `weight`.

`strL` variables are not allowed; see [U] 12.4.8 `strL`.

`markvar()` and `noisily` do not appear in the dialog box.

## Options

### Main

`totals(totals)` and `nototals` control which totals are to be displayed in the table. By default, all totals are reported.

`totals(totals)` specifies which margin totals to display in the reported table. *totals* can contain variables in *rowspec*, *colspec*, *tabspec*, and their interaction. Interactions can be specified by using the # operator.

`nototals` prevents table from displaying any totals.

### Statistics

`statistic(statspec)` specifies the statistic to be displayed. Frequency statistics, summary statistics, and ratio statistics are available by specifying `statistic(freqstat)`, `statistic(sumstat varlist)`, and `statistic(ratiostat [varlist] [, ratio_options])`, respectively.

`statistic()` may be repeated to request multiple statistics.

`statistic(freqstat)` specifies that frequencies be computed.

<i>freqstat</i>	Definition
<code>frequency</code>	frequency
<code>sumw</code>	sum of weights

`statistic(sumstat varlist)` specifies that summary statistic *sumstat* be computed for the variables in *varlist*.

<i>sumstat</i>	Definition
<code>mean</code>	mean
<code>semean</code>	standard error of the mean
<code>sebinomial</code>	standard error of the mean, binomial
<code>sepoisson</code>	standard error of the mean, Poisson
<code>variance</code>	variance
<code>sd</code>	standard deviation
<code>skewness</code>	skewness
<code>kurtosis</code>	kurtosis
<code>cv</code>	coefficient of variation
<code>svycv</code>	coefficient of variation (svy)
<code>geomean</code>	geometric mean
<code>geosd</code>	geometric standard deviation
<code>count</code>	number of nonmissing values
<code>median</code>	median
<code>p#</code>	#th percentile
<code>q1</code>	first quartile
<code>q2</code>	second quartile
<code>q3</code>	third quartile
<code>iqr</code>	interquartile range
<code>min</code>	minimum value
<code>max</code>	maximum value
<code>range</code>	range
<code>first</code>	first value
<code>last</code>	last value
<code>firstnm</code>	first nonmissing value
<code>lastnm</code>	last nonmissing value
<code>total</code>	total
<code>rawtotal</code>	unweighted total
<code>fvfrequency</code>	frequency of each factor-variable level
<code>fvrawfrequency</code>	unweighted frequency of each factor-variable level
<code>fvproportion</code>	proportion within each factor-variable level
<code>fvrawproportion</code>	unweighted proportion within each factor-variable level
<code>fvpercent</code>	percentage within each factor-variable level
<code>fvrawpercent</code>	unweighted percentage within each factor-variable level

`statistic(ratiostat [varlist] [, ratio_options])` specifies that ratio statistic *ratiostat* be computed. If *varlist* is specified, ratios are computed based on the totals of the specified variables. If *varlist* is not specified, ratios are computed based on frequencies.

<i>ratiostat</i>	Definition
<u>proportion</u>	proportion
<u>percent</u>	percentage
<u>rawproportion</u>	proportion ignoring optionally specified weights
<u>rawpercent</u>	percentage ignoring optionally specified weights

  

<i>ratio_options</i>	Definition
<u>across(<i>cellspec</i>)</u>	percentages or proportions across levels of variables or interactions
<u>total</u>	compute overall percentages or proportions

*cellspec* may contain *rowvars*, *colvars*, *tabvars*, or an interaction between any of these variables. Interactions can be specified by using the # operator.

#### Commands

`command(cmdspec)` specifies the Stata commands from which to collect results. `command()` may be repeated to collect results from multiple commands.

*cmdspec* is [*explist*:] *command* [*arguments*] [, *cmdoptions*]

*explist* specifies which results to collect and report in the table. *explist* may include *result identifiers* and *named expressions*.

*result identifiers* are results stored in `r()` and `e()` by the *command*. For instance, *result identifiers* could be `r(mean)`, `r(C)`, or `e(chi2)`. After estimation commands, *result identifiers* also include the following:

Identifier	Result
<code>_r_b</code>	coefficients or transformed coefficients reported by <i>command</i>
<code>_r_se</code>	standard errors of <code>_r_b</code>
<code>_r_z</code>	test statistics for <code>_r_b</code>
<code>_r_z_abs</code>	absolute value of <code>_r_z</code>
<code>_r_p</code>	<i>p</i> -values for <code>_r_b</code>
<code>_r_lb</code>	lower bounds of confidence intervals for <code>_r_b</code>
<code>_r_ub</code>	upper bounds of confidence intervals for <code>_r_b</code>
<code>_r_ci</code>	confidence intervals for <code>_r_b</code>
<code>_r_crlb</code>	lower bounds of credible intervals for <code>_r_b</code>
<code>_r_crub</code>	upper bounds of credible intervals for <code>_r_b</code>
<code>_r_cri</code>	credible intervals for <code>_r_b</code>
<code>_r_df</code>	degrees of freedom for <code>_r_b</code>

*named expressions* are specified as *name* = *exp*, where *name* may be any valid Stata name and *exp* is an expression, typically an expression that involves one or more *result identifiers*.

An example of a named expression is `sd = sqrt(r(variance))`.

For r-class commands, the default is to include all numeric scalars posted to `r()` in the table results. For e-class commands, the default is to include `_r_b` in the table results.

*command* is any command that follows standard Stata syntax.

*arguments* may be anything so long as they do not include an `if` clause, `in` range, or weight specification.

Any `if` or `in` qualifier and weights should be specified directly with `table`, not within the `command()` option.

*cmdoptions* may be anything supported by *command*.

#### Formats

`nformat(%fmt [results] [, basestyle])` changes the numeric format, such as the number of decimal places, for specified results. If *results* are not specified, the numeric format is changed for all results.

*results* may be any statistic named in option `statistic()` (that is, any *freqstat*, *sumstat*, or *ratio*stat) or may be any name in the `e()` or `r()` results produced by commands specified in option `command()`.

This option is repeatable, and when multiple formats apply to one result, the rightmost specification is applied.

This option does not affect the format of numeric layout variables (*rowspec*, *colspec*, and *tabspec*) or the format of factor variables specified in the `statistic()` option. The default format of these variables is taken from the dataset.

*basestyle* indicates that the format be applied to results that do not already have their own format instead of overriding the format for all results.

`sformat(sfmt [results])` changes the string format for specified results. You can, for instance, add symbols or text to the values reported in the table by modifying the string format.

*sfmt* may contain a mix of text and `%s`. Here `%s` refers to the numeric value that is formatted as specified using `nformat()`. The text will be placed around the numeric values in your table as it is placed around `%s` in this option. For instance, to place parentheses around the percent statistics, you can specify `sformat("(%s)" percent)`.

*results* may be any statistic named in option `statistic()` (that is, any *freqstat*, *sumstat*, or *ratio*stat) or may be any name in the `e()` or `r()` results produced by commands specified in option `command()`.

Two text characters must be specified using a special character sequence if you want them to be displayed in your table. To include `%`, type `%%`. To include `\`, type `\\`. For instance, to place a percent sign following percent statistics, you can specify `sformat("%s%" percent)`.

This option is repeatable, and when multiple formats apply to one result, the rightmost specification is applied.

`cidelimiter(char)` changes the delimiter between confidence interval limits to *char*. The default is `cidelimiter(" ")`, that is, two spaces.

`cridelimiter(char)` changes the delimiter between credible interval limits to *char*. The default is `cridelimiter(" ")`, that is, two spaces.

#### Stars

`stars(starspec)` specifies that stars representing statistical significance be included in the table. *starspec* identifies the result whose values determine significance, which characters should represent each significance level, and where these characters should be displayed in the table. *starspec* is

```
starres [#1 "label1" [#2 "label2" [#3 "label3" [#4 "label4" [#5 "label5"]]]]]]
[, attach(attachres) result dimension starsnoteopts]
```

*starres* is the name of the result whose values determine which characters, typically which number of stars, are to be displayed.

*label1* specifies the characters to be displayed when *starres* < #1.

*label2* specifies the characters to be displayed when *starres* < #2.

*label3* specifies the characters to be displayed when *starres* < #3.

*label4* specifies the characters to be displayed when *starres* < #4.

*label5* specifies the characters to be displayed when *starres* < #5.

`attach(attachres)` specifies the name of the result to which the characters defined by *label1*, ..., *label5* are to be attached. If `attach()` is not specified, a new result named **stars** is created and is automatically added to the table.

`result` and `dimension` control how `collect stars` adds items when labeling significant results. These options are mutually exclusive.

`result` specifies the default behavior, and this option is necessary only if the following `dimension` behavior is in effect and you want to change back to the `result` behavior.

`dimension` specifies that dimension **stars** be added to the collection. Items will be tagged with `stars[value]`, and the labels will be tagged with `stars[label]`. Use this option for layouts where results are to be stacked within columns, and use `new dimension stars` in the column specification of the layout.

*starsnoteopts* control the display and composition of the stars note.

`noshownote` and `shownote` control whether to display the stars note.

`increasing` and `decreasing` control the order of *p*-values in the stars note.

`pvname(string)` specifies a name for the *p*-value in the stars note. The default is `pvname(p)`.

`delimiter(string)` specifies the delimiter between labels in the stars note. The default is `delimiter(",")`.

`nformat(%fmt)` specifies the numeric format for the cutoff values in the stars note. The default is `nformat(%9.0g)`.

`prefix(string)` specifies the prefix for the stars note. The prefix is empty by default.

`suffix(string)` specifies the suffix for the stars note. The suffix is empty by default.

For example, `stars(_r_p 0.01 "***" 0.05 "*" 0.1 "**", attach(_r_b))` could be added to a table of regression results to specify that stars be defined based on the *p*-values in `_r_p` and be attached to the reported coefficients (`_r_b`).

#### Options

`listwise` handles missing values through listwise deletion, meaning that the entire observation is omitted from the sample if any variable specified in a `statistic()` option is missing for that observation. By default, `table` will omit an observation only if all variables specified in all `statistic()` options are missing for that observation.

`missing` specifies that numeric missing values of any variables specified in *rowspec*, *colspec*, or *tabspec* be treated as valid categories. By default, observations with a numeric missing value in any of these variables are omitted.



This option does not apply to factor variables specified with statistics `fvfrequency`, `fvrawfrequency`, `fvproportion`, `fvrawproportion`, `fvpercent`, or `fvrawpercent`.

`showcounts` specifies that `table` report the sample size for each variable specified in option `statistic()`.

`zerocounts` specifies that `table` report a 0 in empty cells for results `count`, `frequency`, `fvfrequency`, and `fvrawfrequency`.

`name(cname)` specifies that a collection named *cname* be associated with the collected statistics and results. The default is `name(Table)`.

`append` specifies that `table` append its collection information into the collection named in `name()`.

`replace` permits `table` to overwrite an existing collection. This option is implied for `name(Table)` when `append` is not specified.

`label(filename)` specifies the *filename* containing the collection labels to use for your table. Labels in *filename* will be loaded for the table, and any labels not specified in *filename* will be taken from the labels defined in `c(collect_label)`. The default is to use only the collection labels set in `c(collect_label)`; see [TABLES] [set collect\\_label](#).

`style(filename [, override])` specifies the *filename* containing the collection styles to use for your table. The default collection styles will be discarded, and only the collection styles in *filename* will be applied.

If you prefer the default collection styles but also want to apply any styles in *filename*, specify `override`. If there are conflicts between the default collection styles and those in *filename*, the ones in *filename* will take precedence.

The default is to use only the collection styles set in `c(table_style)`; see [TABLES] [set table\\_style](#).

The following options are available with `table` but are not shown in the dialog box:

`markvar(newvar)` generates an indicator variable that identifies the observations used in the tabulation.

`noisily` specifies that output from the commands specified in `command()` options be displayed. By default, output from commands is suppressed.

## Remarks and examples

[stata.com](http://www.stata.com)

Remarks are presented under the following headings:

- [Introduction](#)
- [Specifying the table layout](#)
- [Advanced table customization](#)

## Introduction

The `table` command can create many customized tables, ranging from simple one-way tabulations to multiple *n*-way tables with summary statistics and estimation results. `table` can compute and report frequencies, proportions, percentiles, and other summary statistics. It can also run other Stata commands and include their results in the table. This means you can combine the summary statistics computed by `table` with test statistics, correlations, regression coefficients, and other results collected from Stata commands. In addition to building tables with the desired statistics, you can customize them by formatting the values in the table and applying predefined styles and labels that affect how the row headers, column headers, and values are displayed in the table.

`table` can accommodate a variety of layouts. You can define the rows, columns, and even separate tables by levels of categorical variables, statistics, or Stata commands.

If your goal is simply to create a table of estimation results or descriptive statistics, you can use the `etable` and `dtable` commands, respectively. These commands allow you to create these tables and export them to a variety of file types in a single step. However, unlike `table`, these commands create tables with a predefined layout. Therefore, if you want control over the layout or you wish to include a combination of summary statistics, estimation results, and results from other Stata commands, you should use the `table` command.

In the following entries, we provide simplified syntax, examples, and discussion for specialized types of tables that can be created using `table`. If you are interested in creating one of these types of tables, we suggest reading the corresponding entry.

[R] <b>table oneway</b>	One-way tabulation
[R] <b>table twoway</b>	Two-way tabulation
[R] <b>table multiway</b>	Multiway tables
[R] <b>table summary</b>	Table of summary statistics
[R] <b>table hypothesis tests</b>	Table of hypothesis tests
[R] <b>table regression</b>	Table of regression results

All the concepts demonstrated in the entries above can be combined to create tables including combinations of tabulations, summary statistics, hypothesis tests, and regression results.

In this entry, we provide additional information on specifying the table layout and which portions of the layout `table` will automate for you. In addition, we provide resources for customizing the table and exporting the results to your preferred format.

## Specifying the table layout

A table's layout is determined by our row, column, and table dimension specifications. For example, we specify variable names to define the rows and place statistics in the columns, or vice versa. Because we can include so many different statistics, we can specify keywords that we use to identify the results we have collected from commands and the statistics that `table` has calculated.

The syntax for specifying the table layout is

```
table ([rowspec]) ([colspec]) ([tabspec])
```

We refer to *rowspec*, *colspec*, and *tabspec* collectively as the “layout”. For some tables, **keywords** are required in the layout to uniquely identify the values that we want to include in our table. If you omit a necessary keyword from the layout, `table` will fill one in for you.

The rules determining whether a keyword is necessary to uniquely identify values in the table are as follows:

1. If more than one statistic is specified, then `result` is needed in the layout.
2. If more than one variable is specified in option `statistic()` and option `command()` is not specified, then `var` is needed in the layout.
3. If more than one `across()` specification is used for ratio statistics, then `across` is needed in the layout.
4. If option `command()` is specified, then `colname` is needed in the layout. If, in addition, more than one variable is specified in option `statistic()`, then `colname` is needed instead of `var`, which was required in 2.

5. If multiple `command()` options are specified and option `statistic()` is not specified, then `command` is needed in the layout.
6. If both options `command()` and `statistic()` are specified, then `statcmd` is needed in the layout.

If we do not directly specify a necessary keyword in one of *rowspec*, *colspec*, or *tabspec*, the missing keywords will be automatically added to the layout as follows:

1. If the row specification is empty, then put the missing keywords in *rowspec*.
2. If the row specification is not empty but the column specification is empty, then put the missing keywords in *colspec*.
3. If the row and column specifications are not empty but the table specification is empty and if `result` is the only missing keyword and there is only one statistic (`result`), then put `result` in *tabspec*.
4. Otherwise, append the missing keywords to *rowvars*.

Below, we demonstrate how missing keywords are added to the *layout*.

Using `auto.dta`, we create a table with the minimum and maximum mpg for each level of `rep78`. The keyword `result` identifies the statistics we computed. By listing an empty set of parentheses followed by `rep78`, we request that the levels of `rep78` be placed on the columns.

```
. use https://www.stata-press.com/data/r18/auto
(1978 automobile data)
. table () rep78, statistic(min mpg) statistic(max mpg)
```

	Repair record 1978					Total
	1	2	3	4	5	
Minimum value	18	14	12	14	17	12
Maximum value	24	24	29	30	41	41

Based on [rule 1](#), if we request more than one statistic, `result` must be in the layout. Based on [situation 1](#), if the row specification is empty, then the missing keyword will be placed in the row specification. We could have created the same table by typing

```
. table (result) (rep78), statistic(min mpg) statistic(max mpg)
```

Now, let's include multiple variables in our `statistic()` option. We also type `rep78` immediately after `table` to specify that the levels of `rep78` be placed on the rows.

```
. table rep78, statistic(mean mpg price)
```

	Mileage (mpg)	Price
Repair record 1978		
1	21	4564.5
2	19.125	5967.625
3	19.43333	6429.233
4	21.66667	6071.5
5	27.36364	5913
Total	21.28986	6146.043

Because we have more than one variable in the `statistic()` option, then keyword `var` must be in the layout ([rule 2](#)). If we include a row specification but leave the column specification empty, `table` will treat `var` as the column identifier. We could have equivalently typed

```
. table (rep78) (var), statistic(mean mpg price)
```

Next, let's include both a `command()` option and a `statistic()` option with multiple variables in the same table. We want a table with coefficients and means of the independent variables. We use the `command()` option to fit the regression and obtain the means with the `statistic()` option. Now, we need both `colname` and `statcmd` to uniquely identify the values in the table. Let's omit `statcmd` from our command.

```
. table (colname) (result[_r_b mean]),
> command(regress mpg turn trunk) statistic(mean turn trunk)
```

	Coefficient	Mean
Turn circle (ft.)		
regress mpg turn trunk	-.7610113	
Mean		39.64865
Trunk space (cu. ft.)		
regress mpg turn trunk	-.3161825	
Mean		13.75676
Intercept		
regress mpg turn trunk	55.82001	

But based on [situation 4](#), `table` will add `statcmd` to the row specification if we leave it out. So we could have also typed the following to create the same table:

```
. table (colname statcmd) (result[_r_b mean]),
  command(regress mpg turn trunk) statistic(mean turn trunk)
```

This table displays each of the statistics that we requested. If we simply wanted to compute some statistics quickly, it has served its purpose. However, if we wish to share these results with others or include a table in a report, we will want to make some modifications.

## Advanced table customization

`table` allows you to customize the results of your table using the `stars()`, `nformat()`, `sformat()`, `cidelimiter()`, `label()`, and `style()` options. With these, you can add significance stars, change the numeric format, and attach characters such as percent signs or parentheses to values in the table, use a stored set of labels, or use a predefined style. See [\[TABLES\] Predefined styles](#) for more information on selecting a style that adjusts elements of the table such as row header alignment, alignment of values within the cells, and which labels are included in the headers.

Customization can also go beyond the predefined styles and options available to you in the `table` command. `table` stores all of its results in a collection named `Table`. This means that you can use the specialized tools available in the `collect` suite of commands to further customize your table. With `collect`, you can modify specific labels, add borders, change the style of the headers, and the like. Once you have a publication-ready table, you can use `collect export` to export your table to HTML, Word, L<sup>A</sup>T<sub>E</sub>X, PDF, Excel, or another format appropriate for your report.

## Stored results

`table` stores the following in `r()`:

Scalars

`r(N)` number of observations

## Methods and formulas

Variables specified in *rowspec*, *colspec*, and *tabspec* identify groups of observations within the dataset. These groups are represented in the table by cells and cell margins (totals). For a given cell or cell margin, let  $n$  denote the number of observations (**frequency**). Let  $x$  denote the variable on which we want to calculate summary statistics, and let  $x_i$ ,  $i = 1, \dots, n$ , denote an individual observation on  $x$ . **count** is the number of nonmissing values of  $x$ . **first** is  $x_1$  and **last** is  $x_n$ . Let  $a$  be the smallest  $i$  such that  $x_i$  is not missing, and then **firstnm** is  $x_a$ . Let  $b$  be the largest  $i$  such that  $x_i$  is not missing, and then **lastnm** is  $x_b$ .

Let  $v_i$  be the weight, and if no weight is specified, define  $v_i = 1$  for all  $i$ . Let  $v.$  denote the sum of the weights (**sumw**):

$$v. = \sum_{i=1}^n v_i$$

When **aweights** or **pweights** are specified, the normalized weights are given by  $w_i = v_i(n/v.)$  with  $w. = n$ ; otherwise,  $w_i = v_i$  and  $w. = v.$ .

The remaining summary statistics are computed according to the following formulas:

**total**

$$x. = \begin{cases} \sum_{i=1}^n v_i x_i & \text{if pweights} \\ \sum_{i=1}^n w_i x_i & \text{otherwise} \end{cases}$$

**rawtotal**

$$\sum_{i=1}^n x_i$$

**mean**

$$\bar{x} = \frac{1}{w.} \sum_{i=1}^n w_i x_i$$

Define  $m_r$  as the  $r$ th moment about the mean:

$$m_r = \frac{1}{w.} \sum_{i=1}^n w_i (x_i - \bar{x})^r$$

**variance**

$$s^2 = \frac{w.}{w. - 1} m_2 = \frac{1}{w. - 1} \sum_{i=1}^n w_i (x_i - \bar{x})^2$$

**sd** (standard deviation)

$$s = \sqrt{s^2}$$

**semean** (standard error of the mean)

$$\text{se}(\bar{x}) = \frac{s}{\sqrt{w}}$$

**sebinomial** (standard error of the mean, binomial distribution)

$$\sqrt{\frac{\bar{x}(1 - \bar{x})}{w}}$$

**sepoisson** (standard error of the mean, Poisson distribution)

$$\sqrt{\frac{\bar{x}}{w}}$$

When **pweights** are specified, **semean**, **sebinomial**, and **sepoisson** are all computed as

$$\text{se}_{\text{pw}}(\bar{x}) = \sqrt{\frac{n}{n-1} \sum_{i=1}^n \left\{ \frac{v_i}{v} (x_i - \bar{x}) \right\}^2}$$

**skewness**

$$m_3 m_2^{-3/2}$$

**kurtosis**

$$m_4 m_2^{-2}$$

**cv** (coefficient of variation)

$$\frac{s}{\bar{x}}$$

**svycv** (coefficient of variation, survey literature)

$$100 \frac{\text{se}(\bar{x})}{|\bar{x}|}$$

**svycv with pweights**

$$100 \frac{\text{se}_{\text{pw}}(\bar{x})}{|\bar{x}|}$$

geomean (geometric mean)

$$\bar{x}_g = \exp\left(\frac{1}{w.} \sum_{i=1}^n w_i \ln x_i\right)$$

geosd (geometric standard deviation)

$$\exp\left(\sqrt{\frac{1}{w. - 1} \sum_{i=1}^n w_i (\ln x_i - \ln \bar{x}_g)^2}\right)$$

Let  $x_{(i)}$  refer to the  $x$  in ascending order, and let  $w_{(i)}$  refer to the corresponding weights of  $x_{(i)}$ .

minimum

$$x_{(1)}$$

maximum

$$x_{(n)}$$

range

$$x_{(n)} - x_{(1)}$$

To obtain the  $p$ th percentile, which we will denote as  $x_{[p]}$ , let  $P = np/100$  and

$$W_{(i)} = \frac{n}{w.} \sum_{j=1}^i w_{(j)}$$

Find the first index  $i$  such that  $W_{(i)} > P$ . The  $p$ th percentile is then

$$x_{[p]} = \begin{cases} \frac{x_{(i-1)} + x_{(i)}}{2} & \text{if } W_{(i-1)} = P \\ x_{(i)} & \text{otherwise} \end{cases}$$

q1 (first quartile)

$$x_{[25]}$$

q2 (second quartile)

$$x_{[50]}$$

q3 (third quartile)

$$x_{[75]}$$

`iqr` (interquartile range)

$$x_{[75]} - x_{[25]}$$

Let  $f$  be an indicator for a specific level of a factor variable and  $f_i$  denote an individual observation on  $f$ .

`fvfrequency` (frequency of the factor variable's level)

$$\sum_{i=1}^n w_i f_i$$

`fvrawfrequency` (unweighted frequency of the factor variable's level)

$$\sum_{i=1}^n f_i$$

`fvproportion` (proportion of the factor variable's level)

$$\frac{1}{w.} \sum_{i=1}^n w_i f_i$$

`fvrawproportion` (unweighted proportion of the factor variable's level)

$$\frac{1}{n} \sum_{i=1}^n f_i$$

`fvpercent` (percentage of the factor variable's level)

$$\frac{100}{w.} \sum_{i=1}^n w_i f_i$$

`fvrawpercent` (unweighted percentage of the factor variable's level)

$$\frac{100}{n} \sum_{i=1}^n f_i$$

`proportion` is computed from ratios of totals. The numerator is taken from the total for the given cell or cell margin, and the denominator is taken from the total for a cell margin that contains the given cell or cell margin. `percent` is `proportion` multiplied by 100.

`rawproportion` and `rawpercent` are similarly computed using unweighted totals.



## Also see

- [R] [table intro](#) — Introduction to tables of frequencies, summaries, and command results
- [R] [table hypothesis tests](#) — Table of hypothesis tests
- [R] [table multiway](#) — Multiway tables
- [R] [table oneway](#) — One-way tabulation
- [R] [table regression](#) — Table of regression results
- [R] [table summary](#) — Table of summary statistics
- [R] [table twoway](#) — Two-way tabulation
- [TABLES] [Intro](#) — Introduction

Stata, Stata Press, and Mata are registered trademarks of StataCorp LLC. Stata and Stata Press are registered trademarks with the World Intellectual Property Organization of the United Nations. StataNow and NetCourseNow are trademarks of StataCorp LLC. Other brand and product names are registered trademarks or trademarks of their respective companies. Copyright © 1985–2023 StataCorp LLC, College Station, TX, USA. All rights reserved.



For suggested citations, see the FAQ on [citing Stata documentation](#).