table summary — Table of summary statistics

Description

In this entry, we discuss how to use `table` to create a table of summary statistics.

Quick start

Table with the mean of `v1`, `v2`, and `v3` for each category of `a1` and `a2`; rows are defined by categories of `a1` and variables `v1`, `v2`, and `v3`

```
table a1 a2, stat(mean v1 v2 v3)
```

Same as above, but also report standard deviations and suppress the totals; rows are defined by the results for each variable within each category of `a1`

```
table (a1 var result) (a2), stat(mean v1 v2 v3) ///
 stat(sd v1 v2 v3) nototals
```

Table with number of observations in each category of `a2` and `a3`, for each level of `a1`

```
table a1, stat(fvfrequency a2 a3)
```

Same as above, and report percentage of observations in each category

```
table a1, stat(fvfrequency a2 a3) ///
  stat(fvpercent a2 a3)
```

Same as above, and report the percentages with a percent sign, using two decimal places, and enclose them in parentheses

```
table (a1) (var result), stat(fvfrequency a2 a3) ///
  stat(fvpercent a2 a3) ///
  nformat(%5.2f fvpercent) sformat("(%s%%)" fvpercent)
```

Menu

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

Syntax

Basic table of summary statistics

```
table [rowvar] [colvar] [if] [in] [weight], statistic(statspec)
  [statistic(statspec) [...]] [options]
```

Customized table of summary statistics

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

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

keyword	Description
`result`	requested statistics
`var`	variables from `statistic()` option
`across`	index `across()` specifications

options	Description
Main	
`totals(`*totals*`)`	report only the specified totals
`nototals`	suppress the marginal totals
Statistics	
`statistic(`*statspec*`)`	statistic to be reported; default is `statistic(frequency)` when no weights are specified and `statistic(sumw)` otherwise
Formats	
`nformat(`*%fmt* [*results*] [`, basestyle`]`)`	specify numeric format
`sformat(`*sfmt* [*results*]`)`	specify string format
Title	
`title(`*string*`)`	add table title
`titlestyles(`*text_styles*`)`	change table title styles
Notes	
`note(`*string*`)`	add table note
`notestyles(`*text_styles*`)`	change table note styles
Export	
`export(`*filename*`.`*suffix* [`,` *export_opts*]`)`	export table
Options	
`listwise`	use listwise deletion to handle missing values
`missing`	treat numeric missing values of variables in *rowspec*, *colspec*, and *tabspec* like other values
`showcounts`	show sample size for all variables in `statistic()` option
`zerocounts`	report 0 for empty cell counts
`name(`*cname*`)`	collect results into a collection named *cname*
`append`	append results to an existing collection
`replace`	replace results of an existing collection
`label(`*filename*`)`	specify the collection labels
`style(`*filename* [`, override`]`)`	specify the collection style
`markvar(`*newvar*`)`	create *newvar* that identifies observations used in the tabulation

`fweights`, `aweights`, `iweights`, and `pweights` are allowed; see **[U] 11.1.6 weight**.

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

`markvar()` does not appear in the dialog box.

text_styles	Description
`font(`[*fontfamily*][`,` *font_opts*]`)`	specify font style
`smcl(`*smcl*`)`	specify formatting for SMCL files
`latex(`*latex*`)`	specify LaTeX macro
`shading(`*sspec*`)`	set background color, foreground color, and fill pattern

font_opts	Description
`size(`#[*unit*]`)`	specify font size
`color(`*color*`)`	specify font color
`variant(`*variant*`)`	specify font variant and capitalization
[`no`]`bold`	specify whether to format text as bold
[`no`]`italic`	specify whether to format text as italic
[`no`]`strikeout`	specify whether to strike out text
[`no`]`underline`	specify whether to underline text

suffix	*fileformat*	Output format
`docx`	`as(docx)`	Microsoft Word
`html`	`as(html)`	HTML 5 with CSS
`pdf`	`as(pdf)`	PDF
`xlsx`	`as(xlsx)`	Microsoft Excel 2007/2010 or newer
`xls`	`as(xls)`	Microsoft Excel 1997/2003
`tex`	`as(tex)`	LaTeX
`smcl`	`as(smcl)`	SMCL
`txt`	`as(txt)`	plain text
`markdown`	`as(markdown)`	Markdown
`md`	`as(md)`	Markdown

export_opts	Description
`as(`*fileformat*`)`	specify document type
`replace`	overwrite existing file
docx_options	available when exporting to `.docx` files
html_options	available when exporting to `.html` files
pdf_options	available when exporting to `.pdf` files
excel_options	available when exporting to `.xls` and `.xlsx` files
tex_options	available when exporting to `.tex` files
smcl_option	available when exporting to `.smcl` files
txt_option	available when exporting to `.txt` files
md_option	available when exporting to `.markdown` and `.md` files

docx_options	Description
`noisily`	show the `putdocx` commands used to export to the Microsoft Word file
`dofile(`*filename*[`, replace`]`)`	save the `putdocx` commands used for exporting to the named do-file

html_options	Description
`append`	append to an existing file
`tableonly`	export only the table to the specified file
`cssfile(`*cssfile*`)`	define the styles in *cssfile* instead of *filename*
`prefix(`*prefix*`)`	use *prefix* to identify style classes

pdf_options	Description
`noisily`	show the `putpdf` commands used to export to the PDF file
`dofile(`*filename*`[, replace])`	save the `putpdf` commands used for exporting to the named do-file

excel_options	Description
`noisily`	show the `putexcel` commands used to export to the Excel file
`dofile(`*filename*`[ , replace])`	save the `putexcel` commands used for exporting to the named do-file
`sheet(`*sheetname*`[ , replace])`	specify the worksheet to use; the default sheet name is `Sheet1`
`cell(`*cell*`)`	specify the Excel upper-left cell as the starting position to export the table; the default is `cell(A1)`
`modify`	modify Excel file
`noopen`	do not open Excel file in memory

`noopen` does not appear in the dialog box.

tex_options	Description
`append`	append to an existing file
`tableonly`	export only the table to the specified file

smcl_option	Description
`append`	append to an existing file

txt_option	Description
`append`	append to an existing file

md_option	Description
`append`	append to an existing file

fontfamily specifies a valid font family.

unit may be `in` (inch), `pt` (point), or `cm` (centimeter). An inch is equivalent to 72 points and 2.54 centimeters. The default is `pt`.

variant may be `allcaps`, `smallcaps`, or `normal`.

`variant(allcaps)` changes the text to all uppercase letters; applicable when publishing items from a collection to Microsoft Word, PDF, LaTeX, and HTML files.

`variant(smallcaps)` changes the text to use large capitals for uppercase letters and smaller capitals for lowercase letters; applicable when publishing items from a collection to Microsoft Word, LaTeX, and HTML files.

`variant(normal)` changes the font variant back to normal and leaves the capitalization unchanged from the original text; applicable when publishing items from a collection to Microsoft Word, PDF, LaTeX, and HTML files.

smcl specifies the name of the SMCL directive to render text for SMCL output. The supported SMCL directives are `input`, `error`, `result`, and `text`.

latex specifies the name of a LaTeX macro to render text for LaTeX output. Example LaTeX macro names are `textbf`, `textsf`, `textrm`, and `texttt`. Custom LaTeX macros are also allowed. If *text* is to be rendered in a cell, title, or note, then *latex* is translated to the following when you export to LaTeX:

latex {*text*}

sspec is

[`background(`*bgcolor*`) foreground(`*fgcolor*`) pattern(`*fpattern*`)`]

bgcolor specifies the background color.

fgcolor specifies the foreground color.

fpattern specifies the fill pattern. A complete list of fill patterns is shown in the *Appendix*.

bgcolor, *fgcolor*, and *color* may be one of the colors listed in the *Appendix*; a valid RGB value in the form ### ### ###, for example, `171 248 103`; or a valid RRGGBB hex value in the form ######, for example, `ABF867`.

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

freqstat	Definition
`frequency`	frequency
`sumw`	sum of weights

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

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

ratiostat	Definition
`proportion`	proportion
`percent`	percentage
`rawproportion`	proportion ignoring optionally specified weights
`rawpercent`	percentage ignoring optionally specified weights

ratio_options	Definition
`across(`*cellspec*`)`	percentages or proportions across levels of variables or interactions
`total`	compute overall percentages or proportions

cellspec may contain any variables in *rowspec*, *colspec*, *tabspec*, or an interaction between any of these variables. Interactions can be specified by using the `#` operator.

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 *ratiostat*).

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 *ratiostat*).

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.

Title

`title(`*string*`)` adds the text *string* as a title to the table.

`titlestyles(`*text_styles*`)` changes the style for the table title. *text_styles* are the following:

`font(`[*fontfamily*] [`, size(#` [*unit*]`) color(`*color*`) variant(`*variant*`)` [`no`]`bold` [`no`]`italic` [`no`]`strikeout` [`no`]`underline`]`)` specifies the font style. These font style properties are applicable when exporting the table to Microsoft Word, Microsoft Excel, PDF, HTML, and LaTeX files, unless otherwise specified.

fontfamily specifies a valid font family. This font style property is applicable when publishing items from a collection to Microsoft Word, Microsoft Excel, PDF, and HTML files.

`size(#` [*unit*]`)` specifies the font size as a number optionally followed by units. This font style property is applicable when publishing items from a collection to Microsoft Word, Microsoft Excel, PDF, and HTML files.

`color(`*color*`)` specifies the text color.

`variant(`*variant*`)` specifies the font variant and capitalization.

`bold` and `nobold` specify the font weight. `bold` changes the font weight to bold; `nobold` changes the font weight back to normal.

`italic` and `noitalic` specify the font style. `italic` changes the font style to italic; `noitalic` changes the font style back to normal.

`strikeout` and `nostrikeout` specify whether to add a strikeout mark to the title. `strikeout` adds a strikeout mark to the title; `nostrikeout` changes the title back to normal.

`underline` and `nounderline` specify whether to underline the table title. `underline` adds a single line under the title; `nounderline` removes the underline.

Only one of `strikeout` or `underline` is allowed when publishing to HTML files.

`smcl(`*smcl*`)` specifies how to render the table title for SMCL output. This style property is applicable only when publishing items from a collection to a SMCL file.

`latex(`*latex*`)` specifies how to render the table title for LaTeX output. This style property is applicable only when publishing items from a collection to a LaTeX file.

`shading(`*sspec*`)` sets the background color, foreground color, and fill pattern. The background color is applicable when exporting the table to Microsoft Word, Microsoft Excel, PDF, HTML, and LaTeX files. The foreground color and fill pattern are applicable when exporting the table to Microsoft Word and Microsoft Excel.

Notes

`note(`*string*`)` adds the text *string* as a note to the table. `note()` may be specified multiple times to add multiple notes. Each note is placed on a new line.

`notestyles(`*text_styles*`)` changes the style for the table notes. *text_styles* are the following:

`font(`[*fontfamily*] [`, size(#` [*unit*]`)` `color(`*color*`)` `variant(`*variant*`)` [`no`]`bold` [`no`]`italic` [`no`]`strikeout` [`no`]`underline`]) specifies the font style. These font style properties are applicable when exporting the table to Microsoft Word, Microsoft Excel, PDF, HTML, and LaTeX files, unless otherwise specified.

fontfamily specifies a valid font family. This font style property is applicable when publishing items from a collection to Microsoft Word, Microsoft Excel, PDF, and HTML files.

`size(#` [*unit*]`)` specifies the font size as a number optionally followed by units. This font style property is applicable when publishing items from a collection to Microsoft Word, Microsoft Excel, PDF, and HTML files.

`color(`*color*`)` specifies the text color.

`variant(`*variant*`)` specifies the font variant and capitalization.

`bold` and `nobold` specify the font weight. `bold` changes the font weight to bold; `nobold` changes the font weight back to normal.

`italic` and `noitalic` specify the font style. `italic` changes the font style to italic; `noitalic` changes the font style back to normal.

`strikeout` and `nostrikeout` specify whether to add a strikeout mark to the notes. `strikeout` adds a strikeout mark to the note; `nostrikeout` changes the note back to normal.

`underline` and `nounderline` specify whether to underline the table notes. `underline` adds a single line under the notes; `nounderline` removes the underline.

Only one of `strikeout` or `underline` is allowed when publishing to HTML files.

`smcl(`*smcl*`)` specifies how to render the table notes for SMCL output. This style property is applicable only when publishing items from a collection to a SMCL file.

`latex(`*latex*`)` specifies how to render the table notes for LaTeX output. This style property is applicable only when publishing items from a collection to a LaTeX file.

`shading(`*sspec*`)` sets the background color, foreground color, and fill pattern. The background color is applicable when exporting the table to Microsoft Word, Microsoft Excel, PDF, HTML, and LaTeX files. The foreground color and fill pattern are applicable when exporting the table to Microsoft Word and Microsoft Excel.

Export

`export(`*filename*`.`*suffix*`[ , `*export_opts*`])` exports the table to the specified file. *export_opts* are the following:

`as(`*fileformat*`)` specifies the file format to which the table is to be exported. This option is rarely specified because, by default, `table` determines the format from the suffix of the file being created.

`replace` permits `table` to overwrite an existing file.

`noisily` specifies that `table` show the commands used to export the table to Microsoft Word, Microsoft Excel, and PDF files. The `putdocx`, `putexcel`, or `putpdf` command used to export the table will be displayed.

`dofile(`*filename*`[ , replace])` specifies that `table` save to *filename* the commands used to export the table to Microsoft Word, Microsoft Excel, and PDF files.

If *filename* already exists, it can be overwritten by specifying `replace`. If *filename* is specified without an extension, `.do` is assumed.

`append` specifies that `table` append the table to an existing file.

This option is applicable when you export the table to an HTML, a LaTeX, a SMCL, a `txt`, or a Markdown file. When you export to HTML and LaTeX files, the `append` option implies the `tableonly` option. Furthermore, when you export to HTML files, if the target CSS file already exists, `table` will also append to it.

`tableonly` specifies that only the table be exported to the specified HTML or LaTeX document. By default, `table` produces complete HTML and LaTeX documents.

When you export to an HTML file, if the `cssfile()` option is not specified, a CSS filename is constructed from *filename*, with the extension replaced with `.css`.

`cssfile(`*cssfile*`)` specifies that `table` define the styles in *cssfile* instead of *filename* when you export to HTML.

`prefix(`*prefix*`)` specifies that `table` use *prefix* to identify style classes when you export to HTML.

`sheet(`*sheetname* `[ , replace])` saves to the worksheet named *sheetname*. For more information about this option, see [RPT] **putexcel**.

`cell(`*cell*`)` specifies an Excel upper-left cell as the starting position to publish the table. The default is `cell(A1)`.

`modify` permits `putexcel set` to modify an Excel file. For more information about this option, see [RPT] **putexcel**.

`noopen` prevents `putexcel` from opening the Excel file in memory for modification. It does not appear in the dialog box. For more information about this option, see [RPT] **putexcel**.

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 option is available with `table` but is not shown in the dialog box:

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

Remarks and examples

Remarks are presented under the following headings:

Basic summary statistic tables

The `table` command can be used to compute a variety of summary statistics and display them in a table. Summary statistics can be computed for the full dataset or across levels of one or more categorical variables.

To demonstrate, we use data from the Second National Health and Nutrition Examination Survey (NHANES II) (McDowell et al. 1981) and create a table reporting the mean body mass index (BMI) of individuals across four regions of the USA. We use the `statistic()` option to request that means be computed, and we specify `region` as our row variable for the table. Thus, means are computed for each region separately and for all the regions combined (`Total`).

```
. use https://www.stata-press.com/data/r19/nhanes2l
(Second National Health and Nutrition Examination Survey)
. table region, statistic(mean bmi)
```

	Mean
Region	
NE	25.57535
MW	25.51936
S	25.63317
W	25.42299
Total	25.5376

The mean BMI is very similar across regions. We might want to look at some additional statistics. We can add the minimums and maximums in our table by repeating our `statistic()` option for each statistic; we will use the `stat()` abbreviation.

```
. table region, stat(mean bmi) stat(min bmi) stat(max bmi)
```

	Mean	Minimum value	Maximum value
Region			
NE	25.57535	15.36715	57.10803
MW	25.51936	14.1351	61.1297
S	25.63317	12.3856	55.43552
W	25.42299	15.69046	54.05056
Total	25.5376	12.3856	61.1297

If we want to include even more statistics, the table will become very wide. We can move the statistics to the rows of our table by specifying the keyword `result` in the first set of parentheses. We place `region` on the columns by specifying this variable in the second set of parentheses.

```
. table (result) (region),
> stat(mean bmi) stat(median bmi) stat(sd bmi)
> stat(min bmi) stat(max bmi)
```

	Region				
	NE	MW	S	W	Total
Mean	25.57535	25.51936	25.63317	25.42299	25.5376
Median	25.00623	24.71567	24.98451	24.66734	24.81812
Standard deviation	4.72798	4.905965	5.084678	4.883534	4.914969
Minimum value	15.36715	14.1351	12.3856	15.69046	12.3856
Maximum value	57.10803	61.1297	55.43552	54.05056	61.1297

Instead of computing many statistics for one variable, we might want to compute one statistic for multiple variables. To do this, we can include a list of variables within a single `statistic()` option. Let's compute the means of age, BMI, and systolic blood pressure (`bpsystol`).

```
. table (result) (region), stat(mean age bmi bpsystol)
```

	Region				
	NE	MW	S	W	Total
Age (years)	47.81584	46.52776	48.19068	47.83828	47.57965
Body mass index (BMI)	25.57535	25.51936	25.63317	25.42299	25.5376
Systolic blood pressure	131.3836	130.4863	131.1626	130.5936	130.8817

Classic Table 1

In many reports, the first discussion of the data is accompanied by a "Table 1", a reporting of summary statistics for all variables of interest. Often the table includes a mixture of continuous and categorical variables. We may also want to specify these in a particular order based on importance. In many cases, the table has multiple columns with the summary statistics reported for each level of a categorical variable of interest.

Here we will demonstrate how to create one variety of such a table. We have two factor variables of interest, `diabetes` and `hlthstat`, for which we would like to compute the percentage of individuals in each category. We will use the `fvpercent` statistic to obtain these percentages. We also have three continuous variables, `age`, `bmi`, and `bpsystol`, for which we would like to compute means. We specify the `statistic()` options in the order we wish to see the results in the table. To specify that the variables in the `statistic()` options appear on the rows, we include the keyword `var` in the first set of parentheses. We place `region` on the columns by listing it in the second set of parentheses.

```
. table (var) (region),
> statistic(fvpercent diabetes)
> statistic(mean age bmi)
> statistic(fvpercent hlthstat)
> statistic(mean bpsystol)
```

	Region				
	NE	MW	S	W	Total
Diabetes status=Not diabetic					
Factor-variable percent	95.32	95.49	94.36	95.62	95.18
Diabetes status=Diabetic					
Factor-variable percent	4.68	4.51	5.64	4.38	4.82
Age (years)					
Mean	47.81584	46.52776	48.19068	47.83828	47.57965
Body mass index (BMI)					
Mean	25.57535	25.51936	25.63317	25.42299	25.5376
Health status=Excellent					
Factor-variable percent	26.95	26.33	19.14	21.68	23.29
Health status=Very good					
Factor-variable percent	26.76	26.01	22.82	25.18	25.07
Health status=Good					
Factor-variable percent	30.26	26.52	28.29	29.14	28.43
Health status=Fair					
Factor-variable percent	12.33	15.12	18.65	17.60	16.16
Health status=Poor					
Factor-variable percent	3.69	6.02	11.11	6.40	7.05
Systolic blood pressure					
Mean	131.3836	130.4863	131.1626	130.5936	130.8817

We have the statistics we want, but clearly our table could be improved. Let's start by applying one of the predefined styles, `table-1`, by adding the `style()` option.

```
. table (var) (region),
> statistic(fvpercent diabetes)
> statistic(mean age bmi)
> statistic(fvpercent hlthstat)
> statistic(mean bpsystol) style(table-1)
```

	Region				
	NE	MW	S	W	Total
Diabetes status					
Not diabetic	95.32	95.49	94.36	95.62	95.18
Diabetic	4.68	4.51	5.64	4.38	4.82
Age (years)	47.81584	46.52776	48.19068	47.83828	47.57965
Body mass index (BMI)	25.57535	25.51936	25.63317	25.42299	25.5376
Health status					
Excellent	26.95	26.33	19.14	21.68	23.29
Very good	26.76	26.01	22.82	25.18	25.07
Good	30.26	26.52	28.29	29.14	28.43
Fair	12.33	15.12	18.65	17.60	16.16
Poor	3.69	6.02	11.11	6.40	7.05
Systolic blood pressure	131.3836	130.4863	131.1626	130.5936	130.8817

This style removes the labels for the type of statistic being reported, cleans up the reporting of factor variables in the row headers, and right-aligns the content in the row headers. In addition, we may want to specify that the means be reported to two decimal places using the `nformat(%6.2f mean)` option.

```
. table (var) (region),
> statistic(fvpercent diabetes)
> statistic(mean age bmi)
> statistic(fvpercent hlthstat )
> statistic(mean bpsystol) style(table-1) nformat(%6.2f mean)
```

	Region				
	NE	MW	S	W	Total
Diabetes status					
Not diabetic	95.32	95.49	94.36	95.62	95.18
Diabetic	4.68	4.51	5.64	4.38	4.82
Age (years)	47.82	46.53	48.19	47.84	47.58
Body mass index (BMI)	25.58	25.52	25.63	25.42	25.54
Health status					
Excellent	26.95	26.33	19.14	21.68	23.29
Very good	26.76	26.01	22.82	25.18	25.07
Good	30.26	26.52	28.29	29.14	28.43
Fair	12.33	15.12	18.65	17.60	16.16
Poor	3.69	6.02	11.11	6.40	7.05
Systolic blood pressure	131.38	130.49	131.16	130.59	130.88

Let's go one step further. Perhaps we want the mean and standard deviation of each continuous variable, and we want the frequency and percent for each factor variable. We need to specify a few more `statistic()` options.

```
. table (var) (region),
> stat(fvfreq diabetes) statistic(fvpercent diabetes)
> statistic(mean age bmi) statistic(sd age bmi)
> statistic(fvfreq hlthstat) statistic(fvpercent hlthstat)
> statistic(mean bpsystol) statistic(sd bpsystol)
> style(table-1) nformat(%6.2f mean sd)
```

	Region				
	NE	MW	S	W	Total
Diabetes status					
Not diabetic	1,997	2,648	2,692	2,513	9,850
	95.32	95.49	94.36	95.62	95.18
Diabetic	98	125	161	115	499
	4.68	4.51	5.64	4.38	4.82
Age (years)	47.82	46.53	48.19	47.84	47.58
	17.02	17.38	16.86	17.53	17.21
Body mass index (BMI)	25.58	25.52	25.63	25.42	25.54
	4.73	4.91	5.08	4.88	4.91
Health status					
Excellent	562	730	546	569	2,407
	26.95	26.33	19.14	21.68	23.29
Very good	558	721	651	661	2,591
	26.76	26.01	22.82	25.18	25.07
Good	631	735	807	765	2,938
	30.26	26.52	28.29	29.14	28.43
Fair	257	419	532	462	1,670
	12.33	15.12	18.65	17.60	16.16
Poor	77	167	317	168	729
	3.69	6.02	11.11	6.40	7.05
Systolic blood pressure	131.38	130.49	131.16	130.59	130.88
	24.31	22.50	24.21	22.42	23.33

Finally, to distinguish among our statistics, we can use the `sformat()` option to add parentheses around our standard deviations and percent signs to our percentages.

```
. table (var) (region),
> stat(fvfreq diabetes) statistic(fvpercent diabetes)
> statistic(mean age bmi) statistic(sd age bmi)
> statistic(fvfreq hlthstat) statistic(fvpercent hlthstat)
> statistic(mean bpsystol) statistic(sd bpsystol)
> style(table-1) nformat(%6.2f mean sd)
> sformat("(%s)" sd) sformat("%s%%" fvpercent)
```

	Region				
	NE	MW	S	W	Total
Diabetes status					
Not diabetic	1,997	2,648	2,692	2,513	9,850
	95.32%	95.49%	94.36%	95.62%	95.18%
Diabetic	98	125	161	115	499
	4.68%	4.51%	5.64%	4.38%	4.82%
Age (years)	47.82	46.53	48.19	47.84	47.58
	(17.02)	(17.38)	(16.86)	(17.53)	(17.21)
Body mass index (BMI)	25.58	25.52	25.63	25.42	25.54
	(4.73)	(4.91)	(5.08)	(4.88)	(4.91)
Health status					
Excellent	562	730	546	569	2,407
	26.95%	26.33%	19.14%	21.68%	23.29%
Very good	558	721	651	661	2,591
	26.76%	26.01%	22.82%	25.18%	25.07%
Good	631	735	807	765	2,938
	30.26%	26.52%	28.29%	29.14%	28.43%
Fair	257	419	532	462	1,670
	12.33%	15.12%	18.65%	17.60%	16.16%
Poor	77	167	317	168	729
	3.69%	6.02%	11.11%	6.40%	7.05%
Systolic blood pressure	131.38	130.49	131.16	130.59	130.88
	(24.31)	(22.50)	(24.21)	(22.42)	(23.33)

We have added many customizations to our table. However, you may prefer a different look. For another style, you can select from the predefined styles described in [TABLES] **Predefined styles**. If none of these provide the exact style you want for your table, you can further customize the results by using the `collect` suite of commands. To learn more, see [TABLES] **Intro**.

If you wish to include this table in a paper, on a webpage, or in another format, you can easily export it in LaTeX, Word, Excel, HTML, and a variety of other formats by using the `export()` option.

Stored results

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

Scalars

`r(N)` number of observations

References

Huber, C. 2021a. Customizable tables in Stata 17, part 3: The classic table 1. *The Stata Blog: Not Elsewhere Classified.* https://blog.stata.com/2021/06/24/customizable-tables-in-stata-17-part-3-the-classic-table-1/.

———. 2021b. Customizable tables in Stata 17, part 4: Table of statistical tests. *The Stata Blog: Not Elsewhere Classified.* https://blog.stata.com/2021/08/24/customizable-tables-in-stata-17-part-4-table-of-statistical-tests/.

McDowell, A., A. Engel, J. T. Massey, and K. Maurer. 1981. "Plan and operation of the Second National Health and Nutrition Examination Survey, 1976–1980". In *Vital and Health Statistics*, ser. 1, no. 15. Hyattsville, MD: National Center for Health Statistics.

Also see

[R] **dtable** — Create a table of descriptive statistics

[R] **table intro** — Introduction to tables of frequencies, summaries, and command results

[R] **tabstat** — Compact table of summary statistics

[TABLES] **Intro** — Introduction