### table regression — Table of regression results

Description	Quick start	Menu	Syntax
Options	Remarks and examples	Stored results	References
Also see			

## Description

In this entry, we discuss how to create tables of regression results. These tables can include coefficients, standard errors, confidence intervals, and many more results stored by estimation commands.

# **Quick start**

Table of regression coefficients; rows correspond to covariates (colname)

table colname, command(regress y x1 x2 x3)

Table of coefficients and confidence intervals; columns correspond to the statistics (result)

table (colname) (result), command(\_r\_b \_r\_ci: regress y x1 x2 x3)

Same as above, but use the labels defined in mylabels.stjson and the styles in mystyle.stjson
table (colname) (result), ///
 command(\_r\_b \_r\_ci: regress y x1 x2 x3) ///
 label(mylabels) style(mystyle)

## Menu

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

## **Syntax**

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

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

keyword	Description
result	requested statistics
stars	stars denoting statistical significance
command	index option command()
colname	column names for matrix statistics
rowname	row names for matrix statistics
coleq	column equation names for matrix statistics
roweq	row equation names for matrix statistics

options	Description
Commands command( <i>cmdspec</i> )	collect results from the specified Stata command
<pre>Formats nformat(%fmt [results][, basestyle]) sformat(sfmt [results]) cidelimiter(char) cridelimiter(char)</pre>	specify numeric format specify string format use character as delimiter for confidence interval limits use character as delimiter for credible interval limits
Stars stars( <i>starspec</i> )	add stars to denote statistical significance
Title title( <i>string</i> ) <u>titlesty</u> les( <i>text_styles</i> )	add table title change table title styles
Notes note( <i>string</i> ) notestyles( <i>text_styles</i> )	add table note change table note styles
<pre>Export export(filename.suffix[, export_opts])</pre>	export table
Options <u>miss</u> ing name( <i>cname</i> ) append replace label( <i>filename</i> ) style( <i>filename</i> [, override])	treat numeric missing values like other values collect results into a collection named <i>cname</i> append results to an existing collection replace results of an existing collection specify the collection labels specify the collection style
noisily	display output from each command

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

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

noisily does not appear in the dialog box.

text_styles	Description
<pre>font([fontfamily][, font_opts])</pre>	specify font style
<pre>smcl(smcl)</pre>	specify formatting for SMCL files
latex( <i>latex</i> )	specify LATEX macro
<pre>shading(sspec)</pre>	set background color, foreground color, and fill pattern
for the sector	Description

<i>J</i> • · · · <b>_</b> • <i>P</i> · · ·	_ ·····F ·····	
<pre>size(#[unit])</pre>	specify font size	
color( <i>color</i> )	specify font color	
<pre>variant(variant)</pre>	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)	ЪТ <sub>Е</sub> Х
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	
<u>noi</u> sily	show the putdocx commands used to export to the Microsoft Word file	
<pre>dofile(filename[, replace])</pre>	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( <i>cssfile</i> )	define the styles in <i>cssfile</i> instead of <i>filename</i>	
prefix( <i>prefix</i> )	use <i>prefix</i> to identify style classes	
pdf_options	Description	
noisily	show the putpdf commands used to export to the PDF file	
<pre>dofile(filename[, replace])</pre>	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	
<pre>dofile(filename[, replace])</pre>	save the putexcel commands used for exporting to the named do-file	
<pre>sheet(sheetname[, replace])</pre>	specify the worksheet to use; the default sheet name is Sheet1	
cell( <i>cell</i> )	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:

 $\det \{text\}$ 

#### 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

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

*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. Weights are passed to *command* only if they are specified.

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

Formats

nformat(% fint [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 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*). 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 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("%%%" 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).

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  $\[Mathbb{L}]_X$ , a SMCL, a txt, or a Markdown file. When you export to HTML and  $\[Mathbb{L}]_X$  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.
- 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

- 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.
- 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:

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

### **Remarks and examples**

Remarks are presented under the following headings:

Introduction Tables with results from a single command Tables with results from multiple estimation commands Regression results with factor variables

#### Introduction

The table command allows us to create tables of regression results. You can create a table that reports coefficients, standard errors, test statistics, confidence intervals, and other statistics from a single model or a table that compares results from multiple models.

table does not fit models directly. Rather, table will run any Stata command that you include in its command() option and place results from that command into the table. You determine which results you would like to see in the table. You can select any of the results stored by the command.

You can also create a table of regression results with etable. However, etable will create tables only with active estimation results, results from margins, or results stored with estimates store. If you are working with any of these results, you can use etable to create and export a table of regression results. However, if you want to include results from other commands, you should use the table command.

#### Tables with results from a single command

We have data from the Second National Health and Nutrition Examination Survey (NHANES II) (Mc-Dowell et al. 1981). The data contain some demographic information, such as the participants' age. The data also contain some measures of health, including the individual's weight, systolic blood pressure (bpsystol), and whether the individual has diabetes.

Here we will create a table with results from a linear regression model for systolic blood pressure as a function of age and weight. We type the command to fit the model in the command() option. In the first set of parentheses following table, we specify that we want the rows to correspond to the levels of colname—this is how we refer to the list of covariates in our regression model. In the second set of parentheses, we specify that we want the columns to correspond to the statistics (result).

```
. use https://www.stata-press.com/data/r19/nhanes21
(Second National Health and Nutrition Examination Survey)
. table (colname) (result), command(regress bpsystol age weight)
Coefficient
Age (years) .6379892
Weight (kg) .4069041
Intercept 71.27096
```

Our table is fairly simple. By default, table includes only the reported coefficients when an estimation command is specified in the command() option.

The table command can easily be used to compare results across groups in our data. For instance, if we want to fit the same model for males and females, we can add sex to our column specification.

table	(colname)	(sex	result)	),	command(regress	bpsystol	age	weight)	)
-------	-----------	------	---------	----	-----------------	----------	-----	---------	---

	Male	Sex Female	Total
Age (years) Weight (kg) Intercept	.4789361 .3346106 84.08037	.7735499 .4586108 61.70456	.6379892 .4069041 71.27096

We can now easily compare results for males, females, and both together.

We may want to see additional statistics reported. Let's extend our table to include both coefficients and standard errors. We can refer to the reported coefficients using the keyword  $\_r\_b$  and to the reported standard errors as  $\_r\_se$ , and we can list these in the command() option before our regression command. Here we also move result to the first set of parentheses so that coefficients and standard errors will be displayed on separate rows.

```
. table (colname result) (sex),
```

```
> command(_r_b _r_se: regress bpsystol age weight)
```

	Male	Sex Female	Total
Age (years)			
Coefficient	.4789361	.7735499	.6379892
Std. error	.0156578	.0155743	.0111315
Weight (kg)			
Coefficient	.3346106	.4586108	.4069041
Std. error	.0197112	.0182401	.0124786
Intercept			
Coefficient	84.08037	61.70456	71.27096
Std. error	1.74867	1.376067	1.041742

We now have the statistics we want in this table, but we may want to modify the look a bit. table allows us to customize the results in our table in a number of ways. We can use the nformat() option to report all results to two decimal places, and we can use the sformat() option to place parentheses around our standard errors.

```
. table (colname result) (sex),
```

- > command(\_r\_b \_r\_se: regress bpsystol age weight)
- > nformat(%6.2f) sformat("(%s)"\_r\_se)

	Male	Sex Female	Total
Age (years)			
Coefficient	0.48	0.77	0.64
Std. error	(0.02)	(0.02)	(0.01)
Weight (kg)			
Coefficient	0.33	0.46	0.41
Std. error	(0.02)	(0.02)	(0.01)
Intercept			
Coefficient	84.08	61.70	71.27
Std. error	(1.75)	(1.38)	(1.04)

Now that we have the parentheses to distinguish standard errors from coefficients, we may not want to see those labels in the row header. We add the style(table-reg3) option to use the predefined style table-reg3, which hides the names of these statistics, right-aligns the names of the variables in the row headers, center aligns the statistics horizontally within each column, and adds vertical space between variables.

```
. table (colname result) (sex),
> command(_r_b _r_se: regress bpsystol age weight)
> nformat(%05.3f) sformat("(%s)" _r_se) style(table-reg3)
```

	Male	Sex Female	Total
Age (years)	0.479	0.774	0.638
	(0.016)	(0.016)	(0.011)
Weight (kg)	0.335	0.459	0.407
	(0.020)	(0.018)	(0.012)
Intercept	84.080	61.705	71.271
	(1.749)	(1.376)	(1.042)

#### Tables with results from multiple estimation commands

Above, we fit the same model to the full dataset and then to groups of observations within that dataset. We may alternatively want to fit different models and display their results in a single table. To do this, we specify multiple command() options.

```
. table (colname result) (command),
```

> command(\_r\_b \_r\_se: regress bpsystol age weight)

> command(\_r\_b \_r\_se: regress bpsystol age weight iron vitaminc zinc)

> nformat(%6.2f) sformat("(%s)" \_r\_se) style(table-reg3)

	1	2
Age (years)	0.64 (0.01)	0.64 (0.01)
Weight (kg)	0.41 (0.01)	0.40 (0.01)
Serum iron (mcg/dL)		-0.01 (0.01)
Serum vitamin C (mg/dL)		-0.79 (0.36)
Serum zinc (mcg/dL)		-0.05 (0.01)
Intercept	71.27 (1.04)	77.50 (1.75)

We may want to modify this table a bit further. Customization of tables can go beyond the predefined styles and options available to you in the table command. table creates a collection of results that can be used in combination with the collect suite of commands to produce highly customized tables.

If we want to add more descriptive labels for the two models, we can use the collect label levels command to define our new labels. After a change using collect, we can type collect preview to see the results.

```
. collect label levels command 1 "Model 1" 2 "Model 2", modify
```

- . collect style header command, level(label)
- . collect preview

	Model 1	Model 2
Age (years)	0.64 (0.01)	0.64 (0.01)
Weight (kg)	0.41 (0.01)	0.40 (0.01)
Serum iron (mcg/dL)		-0.01 (0.01)
Serum vitamin C (mg/dL)		-0.79 (0.36)
Serum zinc (mcg/dL)		-0.05 (0.01)
Intercept	71.27 (1.04)	77.50 (1.75)

### **Regression results with factor variables**

The examples above included only continuous covariates in the models. When we include factor variables, there are a variety of ways that they can be displayed in the headers of the tables. In [TABLES] **Pre-defined styles**, you will find a number of styles that you can choose from. We demonstrate a few here.

We will start with the table-reg1 style. This style is our default table style, except that it identifies the commands in the headers using values 1, 2, ... rather than labeling them with the full command we typed in the command() option.

```
. table (colname) (command result),
> command(regress bpsystol i.agegrp i.sex weight)
> command(regress bpsystol i.agegrp##i.sex weight)
> style(table-reg1)
```

	1	2
Age group=20-29	0	0
Age group=30-39	1.195226	7808968
Age group=40-49	7.251555	2.749774
Age group=50-59	15.94216	10.43724
Age group=60-69	22.83932	16.53001
Age group=70+	30.46609	23.3076
Sex=Male	0	0
Sex=Female	1.040833	-6.777535
Weight (kg)	.4359741	.4242392
Age group=20-29 # Sex=Male		0
Age group=20-29 # Sex=Female		0
Age group=30-39 # Sex=Male		0
Age group=30-39 # Sex=Female		3.942553
Age group=40-49 # Sex=Male		0
Age group=40-49 # Sex=Female		8.79336
Age group=50-59 # Sex=Male		0
Age group=50-59 # Sex=Female		10.6501
Age group=60-69 # Sex=Male		0
Age group=60-69 # Sex=Female		12.20669
Age group=70+ # Sex=Male		0
Age group=70+ # Sex=Female		13.51823
Intercept	86.71019	91.57774

In some cases, for clarity, it is helpful to see both the factor variables and their levels. The table-reg1 style provides this in the output.

When we have nice value labels on our factor variables, we may want to see only those. The tablereg1-fv1 style removes the extra labels. Our table above also reports zero-valued coefficients for base categories in both the main effects of the factor variables and in their interactions. The table-reg1-fv1 style omits the rows for the base categories in the interactions.

. table (colname) (command result), > command(regress bpsystol i.agegrp i.sex weight)

> command(regress bpsystol i.agegrp##i.sex weight)

> style(table-reg1-fv1)

	1	2
20-29	0	0
30-39	1.195226	7808968
40-49	7.251555	2.749774
50-59	15.94216	10.43724
60-69	22.83932	16.53001
70+	30.46609	23.3076
Male	0	0
Female	1.040833	-6.777535
Weight (kg)	.4359741	.4242392
30-39 # Female		3.942553
40-49 <b>#</b> Female		8.79336
50-59 <b>#</b> Female		10.6501
60-69 <b>#</b> Female		12.20669
70+ # Female		13.51823
Intercept	86.71019	91.57774

Sometimes, the tables are more readable when the row headers are right aligned. We can use the table-reg2-fv1 style in this case. Let's also change the numeric format of all the results so that they report only two decimal places.

```
. table (colname) (command result),
```

- > command(regress bpsystol i.agegrp i.sex weight)
- > command(regress bpsystol i.agegrp##i.sex weight)
- > style(table-reg2-fv1) nformat(%6.2f)

	1	2
20-29	0.00	0.00
30-39	1.20	-0.78
40-49	7.25	2.75
50-59	15.94	10.44
60-69	22.84	16.53
70+	30.47	23.31
Male	0.00	0.00
Female	1.04	-6.78
Weight (kg)	0.44	0.42
30-39 # Female		3.94
40-49 <b>#</b> Female		8.79
50 <b>-</b> 59 <b>#</b> Female		10.65
60-69 <b>#</b> Female		12.21
70+ # Female		13.52
Intercept	86.71	91.58

There are many ways that we can further customize our table using the collect suite of commands. We can add column titles for our models as we did above. In addition, we can use collect style row to specify a character to be used between terms in an interaction.

. collect label levels command 1 "Model 1" 2 "Model 2", modify

- . collect style header command, level(label)
- . collect style row stack, delimiter(" X ")
- . collect preview

	Model 1	Model 2
20-29 30-39	0.00 1.20	0.00 -0.78
40-49 50-59	7.25 15.94	2.75 10.44
60-69 70+	22.84 30.47	16.53 23.31
Male Female	0.00 1.04	0.00 -6.78
Weight (kg)	0.44	0.42
40-49 X Female		8.79
60-69 X Female		10.65 12.21
70+ X Female Intercept	86.71	13.52 91.58

If one of the predefined styles in [TABLES] **Predefined styles** does not suit your needs for factorvariable results (or for any other table customization), you can create your own style. To do this, you will use series of collect style commands, and then you can save the style to use later; see [TABLES] collect style save.

If you wish to include your 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 collect export.

### Stored results

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

Scalars

r(N) number of observations

### References

Huber, C. 2021a. Customizable tables in Stata 17, part 5: Tables for one regression model. The Stata Blog: Not Elsewhere Classified. https://blog.stata.com/2021/08/26/customizable-tables-in-stata-17-part-5-tables-for-one-regressionmodel/.

——. 2021b. Customizable tables in Stata 17, part 6: Tables for multiple regression models. *The Stata Blog: Not Elsewhere Classified.* https://blog.stata.com/2021/09/02/customizable-tables-in-stata-17-part-6-tables-for-multiple-regression-models/.

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] table Table of frequencies, summaries, and command results
- [R] table hypothesis tests Table of hypothesis tests
- [R] table intro Introduction to tables of frequencies, summaries, and command results
- [R] **etable** Create a table of estimation results
- [TABLES] Intro Introduction

Stata, Stata Press, Mata, NetCourse, and NetCourseNow are registered trademarks of StataCorp LLC. Stata and Stata Press are registered trademarks with the World Intellectual Property Organization of the United Nations. StataNow is a trademark of StataCorp LLC. Other brand and product names are registered trademarks or trademarks of their respective companies. Copyright © 1985–2025 StataCorp LLC, College Station, TX, USA. All rights reserved.



For suggested citations, see the FAQ on citing Stata documentation.