Title

stata.com

collect style row — Collection styles for row headers

Description

`collect style row` specifies row header style properties. `collect style row` determines how row headers are constructed, how factor variables are displayed, how duplicates are reported, and how labels wrap or truncate.

Quick start

Stack row header elements in a single column

```
collect style row stack
```

Same as above, and use a colon to separate factor variables from their levels

```
collect style row stack, binder(" : ")
```

Place row header elements in separate columns

```
collect style row split
```

Same as above, and use an x to delimit interaction terms

```
collect style row split, delimiter(" x ")
```

Menu

Statistics > Summaries, tables, and tests > Tables and collections > Build and style table

Syntax

Split row header elements across columns

```
collect style row split [ , options split_options ]
```

Stack row header elements in a single column

```
collect style row stack [ , options stack_options ]
```

options	Description
`name(`*cname*`)`	specify row header styles for collection *cname*
`nodelimiter`	place factor-variable and interaction elements in separate cells without a delimiter
`delimiter(`*delim*`)`	use *delim* to delimit interaction terms composed in a single cell
`atdelimiter(`*atdelim*`)`	use *atdelim* to delimit interaction terms containing the `@` symbol
`bardelimiter(`*bardelim*`)`	use *bardelim* to delimit interaction terms containing the `|` symbol
`binder(`*binder*`)`	use *binder* to separate factor variables from their levels
`nobinder`	do not bind factor variables and their levels
`[no]spacer`	add a blank line between stacked row dimensions

`nobinder` is only allowed with `collect style row stack`.

split_options	Description
`dups(`*dups*`)`	specify how duplicate headers are displayed
`position(`*rowpos*`)`	specify the position of the row header to be filled first
`[no]span`	span row headers into empty row header columns

stack_options	Description
`[no]indent`	indent stacked headers
`length(#)`	specify maximum length for stacked headers
`wrapon(`*wrapon*`)`	specify how to break long headers
`wrap(#)`	specify number of lines to allow for long headers
`truncate(`*truncate*`)`	specify how to truncate headers that do not fit
`[no]abbreviate`	abbreviate long words that do not fit within the specified length

Options

Main

`name(`*cname*`)` specifies the collection to which column header style properties are to be applied. By default, properties are applied to the current collection.

`nodelimiter`, `delimiter()`, `atdelimiter()`, and `bardelimiter()` control how to compose factor-variable and interaction terms in headers.

`nodelimiter` specifies that factor-variable and interaction term elements (matrix stripe elements) be split into separate cells.

`delimiter(`*delim*`)` specifies that factor-variable and interaction term elements (matrix stripe elements) be composed in a single cell.

The variables in an interaction term are composed in a single cell using *delim* as the delimiter.

Factor-variable terms serve as their own dimension nested within the stripe dimensions `coleq`, `colname`, `roweq`, and `rowname`. Option `binder()` controls how levels of factor variables are composed within a single cell.

`atdelimiter(`*atdelim*`)` specifies that *atdelim* be used to delimit interaction terms containing the `@` symbol. This option is applicable when, for example, working with results from `contrast`, `mean`, `proportion`, `ratio`, and `total`.

`bardelimiter(`*bardelim*`)` specifies that *bardelim* be used to delimit interaction terms containing the `|` symbol. This option is applicable when, for example, working with results from `anova` and `manova`.

`binder(`*binder*`)` specifies how to compose levels of factor variables within a single cell.

The binder will be applied as long as the factor variable and its levels are not hidden. Note that the default style used by `collect`, which is `style-default.stjson`, will hide the dimension title from the headers. You can use `collect style header` to specify whether to display the label or name for a dimension and whether to display the label or value for the level of a dimension.

`nobinder` specifies that factor variables should not be bound to their levels. By default, when stacking row headers, factor variables are bound to their levels by an equal sign.

This option is only allowed with `collect style row stack`.

`nospacer` and `spacer` control whether a blank line is added between stacked row dimensions.

`nospacer`, the default, prevents the line from being added.

`spacer` adds the line.

Split options

`dups(`*dups*`)` controls how to handle duplicate header elements. *dups* is one of `repeat`, `first`, or `center`.

`dups(repeat)`, the default, specifies that `collect` repeat duplicate header elements.

`dups(first)` specifies that `collect` hide all duplicate header elements, except the first.

`dups(center)` specifies that `collect` horizontally center duplicate header elements, where the header element spans the duplicate header cell locations. When this style is not supported, such as when exporting to Markdown, `dups(first)` is used instead.

`position(`*rowpos*`)` specifies how split headers are filled in when one or more levels of a dimension occupy more than one cell. This option is used when factor variables are displayed in the row headers. *rowpos* may be `left` or `right`.

`position(left)` is the default and specifies that `collect` fill in row headers starting with the leftmost cell. This will result in some empty cells on the right for unbalanced row dimensions.

`position(right)` specifies that `collect` shift the row header cells to the right so that the cells in the last column are all filled in. This will result in some empty cells on the left for unbalanced row dimensions.

`nospan` and `span` control whether row headers span into empty row header columns. This option is effective only when `position(left)` is in effect.

`span`, the default, specifies that row headers should span into empty row header columns. This helps conserve horizontal space. Otherwise, each column of the row header will be forced to be wide enough to accommodate all the cells.

`nospan` specifies that row headers should not span into empty row header columns.

Stack options

`noindent` and `indent` control indenting of stacked headers.

`indent`, the default, turns on indenting.

`noindent` turns off indenting.

`length(#)` specifies the maximum display length for stacked headers.

Long headers, ones that contain more than # display characters, are broken into multiple rows. Values of # less than 5 are ignored.

If header elements are indented, each indent counts as 2 characters. If # is too small to fit the indented headers, # is increased to accommodate the most indented header. For example, if there is one level of indented headers, and `length(5)` was specified, then # is increased to 7.

By default, there is no limit to the header length.

`wrapon(`*wrapon*`)` specifies how to break long headers. *wrapon* may be `word` or `length`.

`wrapon(word)`, the default, specifies that long headers break at word boundaries.

`wrapon(length)` specifies that headers break based on available space.

`wrap(#)` specifies how many lines to allow when long headers are broken into multiple lines. Headers requiring more than # lines are truncated with ellipses. Values of # less than 1 are ignored.

By default, there is no limit to the number of lines for wrapped headers.

`truncate(`*truncate*`)` specifies how to truncate headers that do not fit within the specified number of lines to wrap. *truncate* may be `tail`, `middle`, or `head`.

`truncate(tail)`, the default, specifies that long headers are truncated at the end.

`truncate(middle)` specifies that long headers are truncated in the middle.

`truncate(head)` specifies that long headers are truncated at the beginning.

`noabbreviate` and `abbreviate` control whether long words are abbreviated when `wrapon(word)` is in effect.

`noabbreviate`, the default, specifies that words that do not fit in the specified length should not be abbreviated.

`abbreviate` specifies that long words be abbreviated if they do not fit in the specified length.

Remarks and examples

stata.com

`collect style row` determines how row headers are constructed, how factor variables are displayed, how duplicates are reported, and how labels wrap or truncate. In the examples that follow, we explore how factor-variable and interaction terms are incorporated in row headers.

▷ Example 1: Working with factor variables

Below, we use data from the Second National Health and Nutrition Examination Survey (NHANES II) (McDowell et al. 1981). We begin by fitting a model for systolic blood pressure as a function of `agegrp` and `sex`. We collect the results, requesting that coefficients (`_r_b`) appear in subsequent tables, and use the `quietly` prefix to suppress the output. Then, we arrange the items in our collection with `collect layout`. We place the variable names on the rows and the statistics (`result`) on the columns:

```
. use https://www.stata-press.com/data/r18/nhanes2
. quietly: collect _r_b: regress bpsystol i.agegrp i.sex
. collect layout (colname) (result)
Collection: default
      Rows: colname
   Columns: result
   Table 1: 9 x 1
```

	Coefficient
20-29	0
30-39	2.916153
40-49	9.603552
50-59	18.38803
60-69	24.18566
70+	30.93702
Male	0
Female	-4.015163
Intercept	119.4303

`collect`'s default style omits the dimension titles, and factor variables get treated as dimensions as well. This is why we see the labels for the levels of the factor variables but not the names of the factor variables.

Below, we specify that we want to see the title for `agegrp`. We also specify that we want to split the row headers across columns. Then, we get a preview of our table:

```
. collect style header agegrp, title(label)
. collect style row split
. collect preview
```

		Coefficient
Age group	20-29	0
Age group	30-39	2.916153
Age group	40-49	9.603552
Age group	50-59	18.38803
Age group	60-69	24.18566
Age group	70+	30.93702
Sex	Male	0
Sex	Female	-4.015163
Intercept		119.4303

By splitting, we have created two columns within our row header, one for variable names (or their labels) and one for the levels of the factor variables.

We do not need to see `Age group` and `Sex` repeated on every row, so we can add the `dups(first)` option to indicate that duplicates should be displayed only the first time they appear.

```
. collect style row split, dups(first)
. collect preview
```

	Coefficient
Age group 20–29	0
30–39	2.916153
40–49	9.603552
50–59	18.38803
60–69	24.18566
70+	30.93702
Sex Male	0
Female	-4.015163
Intercept	119.4303

Now, let's stack all the elements of the row headers into a single column.

```
. collect style row stack
. collect preview
```

	Coefficient
Age group=20–29	0
Age group=30–39	2.916153
Age group=40–49	9.603552
Age group=50–59	18.38803
Age group=60–69	24.18566
Age group=70+	30.93702
Male	0
Female	-4.015163
Intercept	119.4303

By default, when we stack row headers, the titles for the factor variables are bound to their levels by an equal sign, and each bound term is placed in a single cell in the row header. With this binding, we cannot see an effect of stacking the row headers for this simple table.

Continuing with the binders for now, we can specify the `binder()` option to bind the factor variables and their levels using other characters. Here we replace the equal sign with a colon.

```
. collect style row stack, binder(":")
. collect preview
```

	Coefficient
Age group:20–29	0
Age group:30–39	2.916153
Age group:40–49	9.603552
Age group:50–59	18.38803
Age group:60–69	24.18566
Age group:70+	30.93702
Male	0
Female	-4.015163
Intercept	119.4303

This may look better if we stack the levels of factor variables underneath their titles. We can obtain this layout by removing the binder with `nobinder`.

```
. collect style row stack, nobinder
. collect preview
```

	Coefficient
Age group	
20-29	0
30-39	2.916153
40-49	9.603552
50-59	18.38803
60-69	24.18566
70+	30.93702
Sex	
Male	0
Female	-4.015163
Intercept	119.4303

Here we demonstrated the stacked and split row header arrangements using factor variables, but these layouts can also be used to control the look of row headers with other dimensions and when you have multiple row dimensions.

◁

▷ Example 2: Working with interactions

When working with models with interactions, you may also want to specify the delimiter. For example, below we create a new collection called `interaction`, which then becomes the current collection. Then, we fit a model with an interaction between `race` and `sex`, requesting that only the coefficients appear in our tables. We specify the same layout as we did in the previous example:

```
. collect create interaction
(current collection is interaction)
. quietly: collect _r_b: regress bpsystol race##sex
. collect layout (colname) (result)
Collection: interaction
      Rows: colname
   Columns: result
   Table 1: 12 x 1
```

	Coefficient
White	0
Black	.8423655
Other	-2.177732
Male	0
Female	-4.32123
White # Male	0
White # Female	0
Black # Male	0
Black # Female	4.479353
Other # Male	0
Other # Female	.3729767
Intercept	132.8476

Note that by default a # is used to delimit interaction terms. Below, we specify that we want to use an x:

```
. collect style row split, delimiter(" x ")
. collect preview
```

	Coefficient
White	0
Black	.8423655
Other	-2.177732
Male	0
Female	-4.32123
White x Male	0
White x Female	0
Black x Male	0
Black x Female	4.479353
Other x Male	0
Other x Female	.3729767
Intercept	132.8476

Stored results

`collect style row` stores the following in `s()`:

Macros
`s(collection)` name of collection

References

Huber, C. 2021. 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/.

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. *Vital and Health Statistics* 1(15): 1–144.

Also see

[TABLES] **collect query** — Query collection style properties

[TABLES] **collect style header** — Collection styles for hiding and showing header components

[TABLES] **collect style column** — Collection styles for column headers