collect style column — Collection styles for column headers

Description

`collect style column` specifies column header style properties. `collect style column` determines how factor variables are displayed in column headers, how duplicates are reported, whether headers are filled from top to bottom or from bottom to top, and the width and spacing of columns.

Quick start

Specify that repeating headers be displayed only once and centered horizontally

```
collect style column, dups(center)
```

Same as above, and set all the columns to have the same width

```
collect style column, dups(center) width(equal)
```

Use an `x` to delimit interaction terms

```
collect style column, delimiter(" x ")
```

Menu

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

Syntax

```
collect style column [ , options ]
```

options	Description
Main	
`name(`*cname*`)`	specify column 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
`dups(`*dups*`)`	specify how duplicate headers are displayed
`position(`*colpos*`)`	specify the position of the column header to be filled first
SMCL/Text	
`extraspace(`#`)`	specify the number of extra spaces between columns in SMCL and plain text
`width(`*widthspec*`)`	specify how to distribute column widths

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. *binder* will be used to separate factor variables from their levels.

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.

`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(`*colpos*`)` specifies how column 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 column headers. *colpos* may be `top` or `bottom`.

`position(top)` is the default and specifies that `collect` fill in column headers starting with the topmost cell. This will result in some empty cells on the bottom for unbalanced column dimensions.

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

SMCL/Text

`extraspace(#)` specifies extra spaces to pad columns when exporting to SMCL and plain text. The first and middle columns get # extra spaces added on both sides. The last column gets # extra spaces added on the left. The default is `extraspace(0)`.

`width(`*widthspec*`)` specifies how to distribute the column widths for the items. Row header widths are not affected by this option.

`width(asis)`, the default, specifies that column widths be determined separately, with each column being as wide as necessary to accommodate the widest cell contents within that column.

`width(equal)` specifies that the item column widths all be equal to the widest cell contents among the items and column headers in all columns.

Remarks and examples

`collect style column` determines how factor variables are displayed in column headers, how duplicates are reported, whether headers are filled from top to bottom or from bottom to top, and the width and spacing of columns.

In the following examples, we explore some styles for column headers that may be of interest when working with factor variables and interactions.

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. We collect the coefficients (_r_b) 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 (colname) on the columns and the statistics (result) on the rows:

```
. use https://www.stata-press.com/data/r19/nhanes2
. quietly: collect _r_b: regress bpsystol i.agegrp
. collect layout (result) (colname)
Collection: default
      Rows: result
   Columns: colname
   Table 1: 1 x 7
```

	Age group 20–29	Age group 30–39	Age group 40–49	Age group 50–59	Age group 60–69	Age group 70+	Intercept
Coefficient	0	2.89081	9.597631	18.32889	24.17618	30.82992	117.3466

Instead of having the repeated header for agegrp, let's specify that it be displayed only once and centered horizontally across the columns it applies to. Then, we will get a preview of our table:

```
. collect style column, dups(center)
. collect preview
```

	Age group						Intercept
	20–29	30–39	40–49	50–59	60–69	70+	
Coefficient	0	2.89081	9.597631	18.32889	24.17618	30.82992	117.3466

◁

Example 2: Working with interactions

When working with models that contain interactions, you may want to specify the delimiter for the interaction terms. 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, collecting only the coefficients. To keep the table from becoming too wide, we use collect style cell to format the coefficients to display only two digits after the decimal, and we suppress the display of the base levels.

```
. collect create interaction
(current collection is interaction)
. quietly: collect _r_b: regress bpsystol sex##race
. collect style cell, nformat(%6.2f)
. collect style showbase off
```

Then, we specify the same layout as we did in the previous example:

```
. collect layout (result) (colname)
Collection: interaction
      Rows: result
   Columns: colname
   Table 1: 1 x 6
```

	Sex Female	Race Black	Race Other	Sex Female Race Black	Sex Female Race Other	Intercept
Coefficient	-4.32	0.84	-2.18	4.48	0.37	132.85

Instead of having the levels of `sex` and `race` in separate cells, we may prefer to place them in a single cell, delimited by an `x`. We make that change below and center the results horizontally:

```
. collect style column, delimiter(" x ")
. collect style cell result, halign(center)
. collect preview
```

	Female	Black	Other	Female x Black	Female x Other	Intercept
Coefficient	-4.32	0.84	-2.18	4.48	0.37	132.85

One last thing we can do to make this table even better would be to set the columns to have equal widths. Currently, the width of each column is determined by its contents. We will not make the change here, because it would make the table wrap. But you can experiment by adding the `width(equal)` option to the `collect style column` command from above.

◁

▷ Example 3: Binders for factor variables and their levels

For some tables, you may want to present the label for the factor variable and its level in a single cell. For example, in example 1 we may have wanted to display `Age group: 20-29`, `Age group: 30-39`, and such. To make this change, you may be tempted to simply type

```
. collect style column, binder(":")
```

However, you will not see the change applied, because dimension titles are hidden with the default style used by `collect`. Factor variables are treated as their own dimension, so you will not see the title for the factor variables. To obtain headers such as `Age group: 20-29`, first we make the collection `default` the current collection. (Our first example was executed in the collection called `default`.) Then, we need to specify that we want to see the title for the age group dimension; specifically, we want to see its label. Then, we can get a preview of the table.

```
. collect set default
. collect style column, binder(":")
. collect style header agegrp, title(label)
. collect preview
```

We do not include the output here, because the resulting table is rather wide. However, you can run these commands to view the resulting table.

◁

Stored results

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

Macros
`s(collection)` name of collection

Reference

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

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

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

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