Title

stata.com

Example 4 — Table of t test results

Description Remarks and examples Reference Also see

Description

In this example, we demonstrate how to use `collect` to store the results of mean-comparison tests (t tests) for levels of a categorical variable in a collection and how to create a customized table with these results.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Collecting statistics

Below, we use data from the Second National Health and Nutrition Examination Survey (NHANES II) (McDowell et al. 1981). We wish to test whether the mean systolic blood pressure (`bpsystol`) is the same across males and females in each category of race. To perform the test for each level of `race`, we use the `by` prefix. We first create a new collection named `ex4` and then use the `collect` prefix to collect the results from each `ttest` command and store them in the new collection. All results that `ttest` returns in `r()` will be collected, but only the ones we have specified will be automatically included in our table.

```
. use https://www.stata-press.com/data/r17/nhanes2l
(Second National Health and Nutrition Examination Survey)
. collect create ex4
(current collection is ex4)
. quietly: collect r(N_1) r(mu_1) r(N_2) r(mu_2) r(p):
> by race, sort: ttest bpsystol, by(sex)
```

These results are stored in the current collection. We can then use `collect layout` to arrange the items from the collection into a table. We place the levels of `race` on the rows and the results (`result`) on the columns.

```
. collect layout (race) (result)
Collection: ex4
      Rows: race
   Columns: result
   Table 1: 3 x 5
```

(*output omitted*)

The labels for these statistics are automatically included in the table, which makes it very wide. Therefore, we omit the table preview from the output. In the following section, we will format the table to make it ready for publication.

Customizing the table

To finalize our table from the previous section, we will want to label which statistics are for males and females, shorten the labels for the statistics, and display the results with two digits to the right of the decimal.

First, let's work on the labels. The statistics are part of the dimension `result`. We list the labels for the levels of this dimension:

```
. collect label list result
  Collection: ex4
   Dimension: result
       Label: Result
Level labels:
         N_1  Sample size n
         N_2  Sample size n
        df_t  Degrees of freedom
       level  Confidence level
        mu_1  x mean for population 1
        mu_2  x mean for population 2
           p  Two-sided p-value
         p_l  Lower one-sided p-value
         p_u  Upper one-sided p-value
          sd  Combined std. dev.
        sd_1  Standard deviation for first variable
        sd_2  Standard deviation for second variable
          se  Std. error
           t  t statistic
```

We would like to remap the statistics for males to their own dimension and similarly for females. This will allow us to categorize the results under the labels `Males` and `Females`. The levels `N_1` and `mu_1` correspond to males, and the levels `N_2` and `mu_2` correspond to females. We also remap the p-values to their own dimension called `Difference`.

```
. collect remap result[N_1 mu_1] = Males
(6 items remapped in collection ex4)
. collect remap result[N_2 mu_2] = Females
(6 items remapped in collection ex4)
. collect remap result[p] = Difference
(3 items remapped in collection ex4)
```

Then, we use `collect style header` to specify that we want to display the title for the specified dimensions. These titles are suppressed by default. Then, we arrange our items once more with the new dimension names. Again, we place the levels of `race` on the rows, but now we place the dimensions `Males`, `Females`, and `Difference` on the columns.

```
. collect style header Males Females Difference, title(name)
. collect layout (race) (Males Females Difference)
Collection: ex4
      Rows: race
   Columns: Males Females Difference
   Table 1: 3 x 5
```

	Males N_1	Males mu_1	Females N_2	Females mu_2	Difference p
White	4312	132.8476	4753	128.5264	1.78e-19
Black	500	133.69	586	133.8481	.9217363
Other	103	130.6699	97	126.7216	.3098674

Our table looks much better. Next, we will add labels to the statistics. The statistics are levels of the new dimensions that we remapped them to. To modify labels for levels of a dimension, we use `collect label levels`.

```
. collect label levels Males N_1 "N" mu_1 "Mean BP"
. collect label levels Females N_2 "N" mu_2 "Mean BP"
. collect label levels Difference p "p-value"
```

Previously, we saw the column headers `Males` and `Females` being repeated. We would like to display these only once and center them horizontally. We can use `collect style column` to make this change. We also set the columns to have the same width. Then, we center-align all the cells in the table. With `collect style cell`, we can modify all cells in the table or specific cells. For example, we wish to format the means and p-values to display two digits to the right of the decimal. Therefore, we specify the levels of the dimensions we want to apply this format to. Then, we get a preview of our table.

```
. collect style column, dups(center) width(equal)
. collect style cell, halign(center)
. collect style cell Males[mu_1] Females[mu_2] Difference[p], nformat(%5.2f)
. collect preview
```

	Males		Females		Difference
	N	Mean BP	N	Mean BP	p-value
White	4312	132.85	4753	128.53	0.00
Black	500	133.69	586	133.85	0.92
Other	103	130.67	97	126.72	0.31

Finally, we will modify the borders in the table by using `collect style cell`. First, we remove the vertical border. Because we do not want any vertical borders, we do not list any levels of the dimension `border_block` when we specify the `border(right, pattern(nil))` option. Our next `collect style cell` command requires a bit more explanation. With it, we add a horizontal border below `Males` to indicate that the first `N` and `Mean BP` are for males. To target this very specific border, we specify `cell_type[column-header]#Males`. Here `cell_type` refers to cells in different parts of the table. We want to make a change only in the column header. We also want to make this change only for the `Males` dimension. By specifying the `#` between the tags, we direct the change only at the dimension `Male` within the column headers. We can also target the border under `Females` by specifying `cell_type[column-header]#Females`. To this command, we add the `border(bottom, pattern(single))` option to place a single border on the bottom of these cells.

```
. collect style cell border_block, border(right, pattern(nil))
. collect style cell cell_type[column-header]#Males
> cell_type[column-header]#Females, border(bottom, pattern(single))
. collect preview
```

	Males		Females		Difference
	N	Mean BP	N	Mean BP	p-value
White	4312	132.85	4753	128.53	0.00
Black	500	133.69	586	133.85	0.92
Other	103	130.67	97	126.72	0.31

After finalizing our table of results, we can export it to another format with `collect export`.

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

Also see

[TABLES] **collect remap** — Remap tags in a collection

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

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