putdocx collect — Add a table from a collection to an Office Open XML (.docx) file

Description

`putdocx collect` allows you to export a customized table from a collection to a table in the active `.docx` file. A collection contains a set of results that have been collected from one or more Stata commands using the `collect:` prefix or the `collect get` command. With the suite of `collect` commands, you can specify the layout and style of your table and customize the table.

Unlike `putdocx table`, which allows you to create tables and modify them as needed, `putdocx collect` is designed for exporting a table that you have already customized using the `collect` suite of commands. To learn more about creating customized tables, see [TABLES] **Intro**.

`putdocx collect` also allows you to include tables created by `table` and `etable` in your report. The `table` command is a powerful command for producing tabulations, tables of summary statistics, tables of regression results, and more. The `etable` command creates tables with the active estimation results, results from `margins`, and results stored with `estimates store`. `table` and `etable` are unique in that they automatically create a collection. Their results can be further styled using the `collect` commands, or they can be included in your report as is with `putdocx collect`. See [R] **table intro** and [R] **etable** for more information on these commands.

Quick start

Create a table in the document using items from the current collection

```
putdocx collect
```

Same as above, and display the `putdocx` commands used to export to the `.docx` file

```
putdocx collect, noisily
```

Same as above, but instead of displaying the commands, save them to the file `myfile.do`

```
putdocx collect, dofile(myfile.do)
```

Syntax

`putdocx collect` [, *options*]

options	Description
`name(`*cname*`)`	use collection *cname*
`memtable`	keep table in memory rather than add it to document
`noisily`	show the `putdocx` commands used to export to the `.docx` file
`dofile(`*filename*[`, replace`]`)`	save the `putdocx` commands used for exporting to the named do-file
`tablename(`*tablename*`)`	specify a name for the table

Options

`name(`*cname*`)` specifies a collection *cname* from which to export the customized table. By default, the customized table is taken from the current collection.

`memtable` specifies that the table be created and held in memory instead of being added to the active document. By default, the table is added to the document. This option is useful if the table is intended to be added to a cell of another table or to be used multiple times later.

`noisily` specifies that `putdocx collect` show the `putdocx` commands used to export to the `.docx` file.

`dofile(`*filename*`[ , replace ])` specifies that `putdocx collect` save to *filename* the `putdocx` commands used to export to the `.docx` file. If *filename* already exists, it can be overwritten by specifying `replace`. If *filename* is specified without an extension, `.do` is assumed.

`tablename(`*tablename*`)` specifies a name for the table. By naming the table, you can make further edits with `putdocx table`. The name must be a valid name according to Stata's naming conventions; see **[U] 11.3 Naming conventions**.

If the current collection contains multiple tables, the table names will contain the prefix *tablename* and an integer as the suffix. For example, if you specify `tablename(myreg)` and the collection contains three tables, the table names will be `myreg1`, `myreg2`, and `myreg3`. Also, note that a name is a sequence of 1 to 32 letters, digits, and underscores. If you are exporting multiple tables, consider using a shorter sequence to account for the integer suffix.

Remarks and examples

Remarks are presented under the following headings:

Introduction

`putdocx collect` exports a table from a collection to the active `.docx` file. A collection is a set of results that have been obtained from one or more Stata commands with the `collect` suite of commands, from the `table` command, or from the `etable` command.

To create a table with the `collect` commands, you first collect results from one or more Stata commands. Then you create a table by specifying a layout that determines which results are to be included in the table and how they are to be arranged. For example, you can have the rows correspond to variables and the columns correspond to the statistics, or vice versa. You can also modify the labels in the table, format the results, add borders, change the font style, and make other styling changes to the table. Once you have finalized your table, you can add it to your active `.docx` file with `putdocx collect`.

It is possible to have many collections in memory. However, there is only one current collection, the one you are working with. The current collection is the one that will be exported with `putdocx collect`, but you can export a table from another collection by specifying the `name()` option.

A table that is conveniently added to a `.docx` file using `putdocx collect` could equivalently be added using a series of `putdocx table` commands. To see what those commands look like, use the `noisily` option with `putdocx collect`. This long list will quickly fill up your Results window, so

you may want to store those commands in a do-file instead by using the dofile() option with putdocx collect. For reproducibility purposes, you may choose to include your collect and putdocx collect commands in your do-file, or you may instead incorporate the commands from the do-file created by the dofile() option.

While the collect commands provide many generic styling options that will be applied to the table you export to your .docx file, the suite also includes collect style putdocx, which provides styling options specifically for tables exported to the .docx format. You can use collect style putdocx to specify the alignment of the table on the page, adjust column widths relative to the page width, and more.

In the following sections, we demonstrate how to create a customized table and add the table to the active .docx file. In these examples, we assume some familiarity with the table and collect commands. We recommend that you see [R] **table intro** for information on table and [TABLES] **Intro** for more information on collect to learn more about creating and customizing tables before incorporating them into your .docx file.

Export a collection

Suppose we would like to create a table comparing some measures of health between males and females. We will use data from the Second National Health and Nutrition Examination Survey (NHANES II) (McDowell et al. 1981). Our first step will be to collect the statistics we want, then we will specify the layout for our table and make styling edits as needed.

We would like to compare the mean and standard deviation of systolic blood pressure (bpsystol), body mass index (bmi), and weight for males and females. We compute the statistics using table. We specify var in the first set of parentheses so that the variables will appear on the rows. We specify sex and result in the second set of parentheses so that the sex and the statistics will define the columns. We also specify that we want two digits reported after the decimal for each statistic and that we do not want totals to be reported. The table command will display a table with these statistics and store them in a collection.

```
. use https://www.stata-press.com/data/r19/nhanes2l, clear
. table (var) (sex result), statistic(mean bpsystol bmi weight)
> statistic(sd bpsystol bmi weight) nformat(%7.2f) nototals
```

	Sex			
	Male		Female	
	Mean	Standard deviation	Mean	Standard deviation
Systolic blood pressure	132.89	20.99	129.07	25.13
Body mass index (BMI)	25.51	4.02	25.56	5.60
Weight (kg)	77.98	13.64	66.39	14.73

We now have a table with the statistics we want, and we can export it to a .docx file. But let's make a few styling edits before exporting this table. We can format all cells in the collection, all cells in a particular dimension, or specific cells in a particular dimension. Note that our table has three dimensions—var, sex, and result.

First, we can modify a table header using collect style header. The word Sex we see at the top of the table is the title for the dimension sex. table displays the title for this dimension by default. Because the labels Male and Female make this clear, we can specify title(hide) for this header. The full label Standard deviation is fairly long for a column header, so we can shorten it to SD using collect label levels. In this case, we modify only the sd level of the result dimension.

```
. collect style header sex, title(hide)
. collect label levels result sd "SD", modify
```

We now type `collect layout` without any arguments to view the current layout.

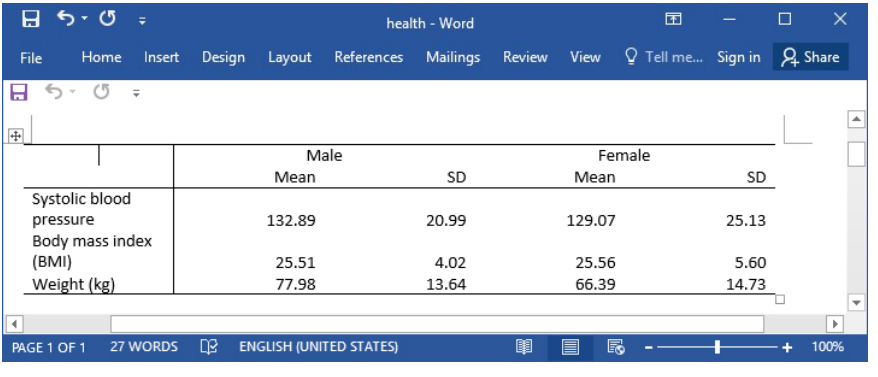
```
. collect layout
Collection: Table
      Rows: var
   Columns: sex#result
   Table 1: 3 x 4
```

	Male		Female	
	Mean	SD	Mean	SD
Systolic blood pressure	132.89	20.99	129.07	25.13
Body mass index (BMI)	25.51	4.02	25.56	5.60
Weight (kg)	77.98	13.64	66.39	14.73

Now that we are done polishing our table of results, we create an active document, export the table, and save our work under the filename `health.docx`.

```
. putdocx begin
. putdocx collect
(collection Table posted to putdocx)
. putdocx save health, replace
successfully created "C:/mypath/health.docx"
```

The table is displayed as follows:



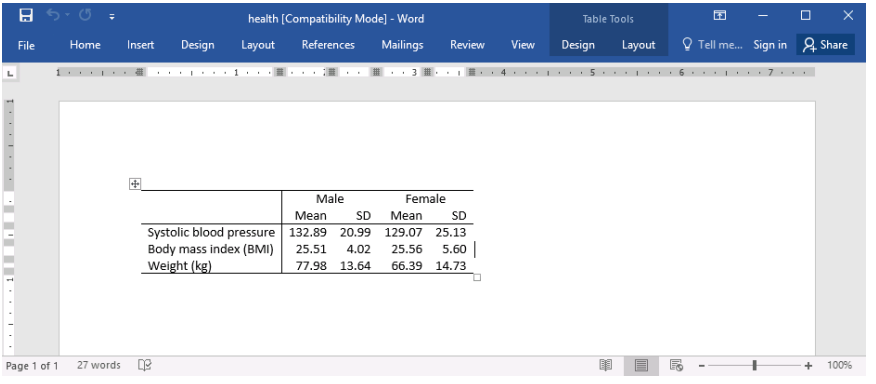
We have a nicely formatted table in our file; however, it is much wider than it needs to be. Also, it would be nice to have the variable labels placed on a single row. In the next section, we demonstrate how to change the width of the table as a whole and the width of the columns.

Specify the style for a collection

To finalize the look for our table from the previous section, we need to use one of the formatting features available with `putdocx table`. We can use `collect style putdocx` to apply some of these formatting options to our collection. We will use the `autofitcontents` layout, which will resize the column widths so that the contents fit.

```
. collect style putdocx, layout(autofitcontents)
. putdocx begin
. putdocx collect
(collection Table posted to putdocx)
. putdocx save health, replace
successfully replaced "C:/mypath/health.docx"
```

Now `health.docx` contains the following:



Overall, our table looks neater, and the variable labels are not spread out across multiple rows.

Stored results

`putdocx collect` stores the following in `s()`:

Macros

`s(collection)`	name of collection
`s(dofile)`	name of the new do-file

References

Huber, C. 2021. Customizable tables in Stata 17, part 2: The new collect command. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2021/06/07/customizable-tables-in-stata-17-part-2-the-new-collect-command/.

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

[RPT] **putdocx intro** — Introduction to generating Office Open XML (.docx) files

[RPT] **putdocx begin** — Create an Office Open XML (.docx) file

[RPT] **putdocx pagebreak** — Add breaks to an Office Open XML (.docx) file

[RPT] **putdocx paragraph** — Add text or images to an Office Open XML (.docx) file

[RPT] **putdocx table** — Add tables to an Office Open XML (.docx) file

[R] **etable** — Create a table of estimation results

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

[TABLES] **Intro** — Introduction

[TABLES] **collect layout** — Specify table layout for the current collection

[TABLES] **collect style putdocx** — Collection styles for putdocx