

# Package ‘crosstable’

November 12, 2023

**Title** Crosstables for Descriptive Analyses

**Version** 0.7.0

**Description** Create descriptive tables for continuous and categorical variables.

Apply summary statistics and counting function, with or without a grouping variable, and create beautiful reports using 'rmarkdown' or 'officer'.

You can also compute effect sizes and statistical tests if needed.

**License** GPL-3

**URL** <https://danchaltiel.github.io/crosstable/>,  
<https://github.com/DanChaltiel/crosstable/>

**BugReports** <https://github.com/DanChaltiel/crosstable/issues/>

**Depends** R (>= 3.1.0)

**Imports** checkmate (>= 1.9.0), cli (>= 3.0.0), dplyr (>= 1.1.0),  
flextable (>= 0.5.1), forcats (>= 1.0.0), glue (>= 1.3.0),  
lifecycle (>= 0.2.0), methods, officer (>= 0.4.0), purrr (>=  
0.2.3), rlang (>= 1.0.0), stats, stringr (>= 1.4.0), tibble (>=  
1.1), tidyr (>= 1.0.0), utils,

**Suggests** callr, covr, crayon, xml2, digest, gt, expss, ggplot2,  
gmodels, Hmisc, hms, jsonlite, knitr, lubridate, openxlsx,  
rmarkdown, sloop, stringi, survival, systemfonts, tidyselect,  
testthat (>= 3.0.0), withr, waldo

**VignetteBuilder** knitr

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.2.3

**Config/testthat/edition** 3

**Config/testthat/parallel** true

**Config/testthat/start-first** by\_factor , effects, officer

**NeedsCompilation** no

**Author** Dan Chaltiel [aut, cre] (<<https://orcid.org/0000-0003-3488-779X>>),  
David Hajage [ccp]

**Maintainer** Dan Chaltiel <dan.chaltiel@gmail.com>

**Repository** CRAN

**Date/Publication** 2023-11-12 20:43:19 UTC

## R topics documented:

apply_labels . . . . .	3
as_gt.crosstable . . . . .	4
as_workbook . . . . .	5
body_add_crosstable . . . . .	6
body_add_crosstable_footnote . . . . .	8
body_add_gg2 . . . . .	8
body_add_img2 . . . . .	9
body_add_legend . . . . .	10
body_add_list . . . . .	13
body_add_normal . . . . .	15
body_add_table_list . . . . .	17
body_add_table_section . . . . .	18
body_add_title . . . . .	19
body_replace_text_at_bkms . . . . .	21
clean_names_with_labels . . . . .	21
confint_numeric . . . . .	22
crosstable . . . . .	23
crosstable_effect_args . . . . .	26
crosstable_options . . . . .	27
crosstable_peek_options . . . . .	31
crosstable_reset_options . . . . .	32
crosstable_test_args . . . . .	32
cross_summary . . . . .	34
ct_compact . . . . .	34
display_effect . . . . .	36
display_test . . . . .	36
docx_bookmarks2 . . . . .	37
effect_summary . . . . .	38
effect_survival . . . . .	39
effect_tabular . . . . .	39
format_fixed . . . . .	40
generate_autofit_macro . . . . .	41
get_label . . . . .	42
get_percent_pattern . . . . .	43
import_labels . . . . .	44
iris2 . . . . .	45
is.crosstable . . . . .	46
mtcars2 . . . . .	47
N . . . . .	48
na . . . . .	48
narm . . . . .	49

*apply\_labels* 3

peek . . . . .	49
pivot_crosstable . . . . .	50
plim . . . . .	50
remove_labels . . . . .	51
rename_with_labels . . . . .	52
set_label . . . . .	52
summaryFunctions . . . . .	53
test_correlation_auto . . . . .	55
test_summarize_auto . . . . .	56
test_summarize_linear_contrasts . . . . .	56
test_survival_logrank . . . . .	57
test_tabular_auto . . . . .	58
transpose_crosstable . . . . .	58
write_and_open . . . . .	59

**Index** 60

---

<code>apply_labels</code>	<i>Batch set variable labels</i>
---------------------------	----------------------------------

---

### Description

This function is a copycat of from `expss` package v0.10.7 (slightly modified) to avoid having to depend on `expss`. See `expss::apply_labels()` for more documentation. Note that this version is not compatible with `data.table`.

### Usage

```
apply_labels(data, ..., warn_missing = FALSE)
```

### Arguments

<code>data</code>	<code>data.frame/list</code>
<code>...</code>	named arguments
<code>warn_missing</code>	if TRUE, throw a warning if some names are missing

### Value

An object of the same type as `data`, with labels

### Author(s)

Dan Chaitiel

**Examples**

```
iris %>%
  apply_labels(Sepal.Length="Length of Sepal",
              Sepal.Width="Width of Sepal") %>%
  crosstable()
```

---

as_gt.crosstable	<i>Converts a crosstable object into a formatted gt table.</i>
------------------	--

---

**Description**

Converts a crosstable object into a formatted gt table.

Method to convert an object to a gt table

Default method to convert an object to a gt table

**Usage**

```
## S3 method for class 'crosstable'
as_gt(
  x,
  show_test_name = TRUE,
  by_header = NULL,
  keep_id = FALSE,
  generic_labels = list(id = ".id", variable = "variable", value = "value", total =
    "Total", label = "label", test = "test", effect = "effect"),
  ...
)

as_gt(x, ...)
```

## Default S3 method:  
as\_gt(x, ...)

**Arguments**

x	object to be converted
show_test_name	in the test column, show the test name
by_header	a string to override the by header
keep_id	whether to keep the .id column
generic_labels	names of the crosstable default columns
...	arguments for custom methods

**Value**

a formatted gt table

**Methods (by class)**

- `as_gt(crosstable)`: For crosstables
- `as_gt(default)`: default function

**Author(s)**

Dan Chaltiel

**See Also**

[as\\_flextable.crosstable\(\)](#)

[gt::gt\(\)](#)

**Examples**

```
xx = mtcars2 %>% dplyr::select(2:10)
crosstable(xx) %>% as_gt
crosstable(xx, by=am) %>% as_gt
crosstable(xx, by=cyl, test=TRUE, total=TRUE) %>%
  as_gt(keep_id=TRUE, show_test_name=FALSE, by_header="Cylinders")
```

---

as\_workbook

*Converts a crosstable object into a formatted, savable openxlsx workbook.*

---

**Description**

Converts a crosstable object into a formatted, savable openxlsx workbook.

**Usage**

```
as_workbook(
  x,
  show_test_name = TRUE,
  by_header = NULL,
  keep_id = FALSE,
  generic_labels = list(id = ".id", variable = "variable", value = "value", total =
    "Total", label = "label", test = "test", effect = "effect"),
  ...
)
```

**Arguments**

**x** the result of `crosstable()` or a list of crosstables  
**show\_test\_name** in the test column, show the test name  
**by\_header** a string to override the by header  
**keep\_id** whether to keep the .id column  
**generic\_labels** names of the crosstable default columns  
**...** unused

**Value**

an openxlsx workbook containing the crosstable(s)

**Author(s)**

Dan Chaltiel

**Examples**

```

library(openxlsx)
target = tempfile(fileext=".xlsx")

x=crosstable(mtcars2, c(mpg, vs, gear), total=TRUE, test=TRUE)
as_workbook(x, keep_id=TRUE) %>%
  saveWorkbook(file=target)
if(interactive()) browseURL(target)

target = tempfile(fileext=".xlsx")
x2=list(iris=crosstable(iris2), crosstable(mtcars2))
as_workbook(x2, keep_id=TRUE) %>%
  saveWorkbook(file=target)
if(interactive()) browseURL(target)

```

---

body\_add\_crosstable *Add a crosstable to an officier document*

---

**Description**

`body_add_crosstable()` adds such a flextable an officier document.

**Usage**

```

body_add_crosstable(
  doc,
  x,
  body_fontsize = NULL,
  header_fontsize = ceiling(body_fontsize * 1.2),
  padding_v = NULL,

```

```
    allow_break = TRUE,  
    max_cols = 25,  
    ...  
  )
```

### Arguments

<code>doc</code>	a rdocx object, created by <code>officer::read_docx()</code>
<code>x</code>	a crosstable object
<code>body_fontsize</code>	fontsize of the body
<code>header_fontsize</code>	fontsize of the header. Defaults to $1.2 \times \text{body\_fontsize}$ .
<code>padding_v</code>	vertical padding of all table rows
<code>allow_break</code>	allow crosstable rows to break across pages
<code>max_cols</code>	max number of columns for <code>x</code>
<code>...</code>	further arguments passed to <code>as_flextable.crosstable()</code>

### Value

The docx object `doc`

### Author(s)

Dan Chaltiel

### Examples

```
#Officer  
library(officer)  
mytable = crosstable(mtcars2)  
doc = read_docx() %>%  
  body_add_crosstable(mytable) %>%  
  body_add_break %>%  
  body_add_crosstable(mytable, compact=TRUE)  
  
dfile = tempfile(fileext=".docx")  
print(doc, target = dfile)  
if(interactive()) browseURL(dfile)
```

---

body\_add\_crosstable\_footnote

*Adds a standard footnote explaining the abbreviations used in a crosstable*

---

### Description

Use it below [body\\_add\\_crosstable\(\)](#). Footnote: Med: median, IQR: interquartile range, Std: standard deviation. Percentages are expressed in column.

### Usage

```
body_add_crosstable_footnote(doc)
```

### Arguments

doc                    a rdocx object

### Value

The docx object doc

### Author(s)

Dan Chaltiel

---

body\_add\_gg2

*Alternative to [officer::body\\_add\\_gg\(\)](#) which uses ggplot syntax*

---

### Description

Alternative to [officer::body\\_add\\_gg\(\)](#) which uses ggplot syntax

### Usage

```
body_add_gg2(
  doc,
  value,
  width = 6,
  height = 5,
  units = getOption("crosstable_units", "in"),
  style = getOption("crosstable_style_image", doc$default_styles$paragraph),
  res = 300,
  ...
)
```



**Arguments**

doc	an rdocx object
value	ggplot object
width, height	width and height. Can be abbreviated to w and h.
units	units for width and height
style	paragraph style
res	resolution of the png image in ppi (passed to the argument dpi of <code>ggplot2::ggsave()</code> )
...	other arguments to be passed to <code>ggplot2::ggsave()</code>

**Value**

The docx object doc

**Author(s)**

Dan Chaltiel

**Examples**

```
library(officer)
library(ggplot2)
p = ggplot(data=iris, aes(Sepal.Length, Petal.Length)) + geom_point()
crosstable_options(
  units="cm",
  style_image="centered"
)
doc = read_docx() %>%
  body_add_normal("Text before") %>%
  body_add_gg2(p, w=14, h=10, scale=1.5) %>% #or units="cm" instead of using options
  body_add_normal("Text after")
write_and_open(doc)
```

---

body\_add\_img2

*Alternative to `officer::body_add_img()` which adds a units choice*

---

**Description**

Alternative to `officer::body_add_img()` which adds a units choice

**Usage**

```
body_add_img2(
  doc,
  src,
  width,
  height,
```

```

units = getOption("crosstable_units", "in"),
style = getOption("crosstable_style_image", doc$default_styles$paragraph),
...
)

```

### Arguments

doc	an rdocx object
src	image filename, the basename of the file must not contain any blank.
width, height	width and height. Can be abbreviated to w and h.
units	units for width and height
style	paragraph style
...	other arguments to be passed to <code>officer::body_add_img()</code>

### Value

The docx object doc

### Author(s)

Dan Chaltiel

### See Also

[body\\_add\\_gg2\(\)](#)

### Examples

```

img.file = file.path( R.home("doc"), "html", "logo.jpg" )
if(file.exists(img.file)){
  library(officer)
  options(crosstable_units="cm")
  doc = read_docx() %>%
    body_add_normal("This is the R logo.") %>%
    body_add_img2(img.file, h=7.6, w=10, style="centered") #or units="cm" without options
  #write_and_open(doc)
}

```

---

body\_add\_legend

*Add a legend to a table or a figure*

---

### Description

Add a legend to a table or a figure in an officer document. Legends can be referred to using the `@ref` syntax in `body_add_normal()` (see examples for some use cases). Table legends should be inserted before the table while figure legends should be inserted after the figure.

**Usage**

```
body_add_table_legend(
    doc,
    legend,
    ...,
    bookmark = NULL,
    legend_style = getOption("crosstable_style_legend", doc$default_styles$paragraph),
    style = deprecated(),
    legend_prefix = NULL,
    name_format = NULL,
    legend_name = "Table",
    seqfield = "SEQ Table \\* Arabic",
    par_before = FALSE,
    legacy = FALSE
)
```

```
body_add_figure_legend(
    doc,
    legend,
    ...,
    bookmark = NULL,
    legend_style = getOption("crosstable_style_legend", doc$default_styles$paragraph),
    style = deprecated(),
    legend_prefix = NULL,
    name_format = NULL,
    legend_name = "Figure",
    seqfield = "SEQ Figure \\* Arabic",
    par_after = FALSE,
    legacy = FALSE
)
```

**Arguments**

doc	a docx object
legend	the table legend. Supports glue syntax and markdown syntax (see Section below).
...	unused
bookmark	the id of the bookmark. This is the id that should then be called in <a href="#">body_add_normal()</a> using the "\\@ref(id)" syntax. Forbidden characters will be removed.
legend_style	style of of the whole legend. May depend on the docx template. However, if name_format is provided with a specific font.size, this size will apply to the whole legend for consistency.
style	deprecated in favor of name_format.
legend_prefix	a prefix that comes before the legend, after the numbering
name_format	format of the legend's LHS (legend_name + numbering) using <a href="#">officer::fp_text_lite()</a> or <a href="#">officer::fp_text()</a> . Default to fp_text_lite(bo1d=TRUE) in addition to

	the format defined in <code>legend_style</code> . Note that the reference to the bookmark will have the same specific format in the text.
<code>legend_name</code>	name before the numbering. Default to either "Table" or "Figure".
<code>seqfield</code>	Keep default. Otherwise, you may figure it out doing this: in a docx file, insert a table legend, right click on the inserted number and select "Toggle Field Codes". This argument should be the value of the field, with extra escaping.
<code>par_before</code> , <code>par_after</code>	should an empty paragraph be inserted before/after the legend?
<code>legacy</code>	use the old version of this function, if you cannot update <code>{officer}</code> to v0.4+

**Value**

The docx object `doc`

**Warning**

Be aware that you unfortunately cannot reference a bookmark more than once using this method.

Writing:

```
body_add_normal("Table \@ref(iris_col1) is about flowers. I really like Table \@ref(iris_col1).")
```

will prevent the numbering from applying.

**What to do if there is still no numbering?**

During the opening of the document, MS Word might ask you to "update the fields", to which you should answer "Yes".

If it is not asked or if you answer "No", the legends added with `body_add_table_legend()` or `body_add_figure_legend()` might have no actual numbers displayed.

In this case, you have to manually update the references in MS Word: select all (Ctrl+A), then update (F9), sometimes twice. More info on <https://ardata-fr.github.io/officeverse/faq.html#update-fields>.

**Markdown support**

In all crosstable helpers for `officer`, you can use the following Markdown syntax to format your text:

- *bold*: `**text in bold**`
- *italics*: `*text in italics*`
- *subscript*: `Text in ~subscript~`
- *superscript*: `Text in ^superscript^`
- *newline*: Before `<br>` After
- *color*: `<color:red>red text</color>`
- *shade*: `<shade:yellow>yellow text</shade>` (background color)
- *font family*: `<ff:symbol>symbol</ff>` (

Note that the font name depends on your system language. For instant, in French, it would be `Symbol` with an uppercase first letter.

See the last example of `body_add_normal()` for a practical case.

**Author(s)**

Dan Chaltiel

**Examples**

```

library(officer)
library(ggplot2)
p = ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species)) + geom_point()
fp_italic = fp_text_lite(italic=TRUE, font.size=10)
x = read_docx() %>%
  body_add_normal("There is Table \@ref(iris_col1) and Table \@ref(iris_col2). ",
                  "The `iris` dataset is about flowers.") %>%
  body_add_normal() %>%
  body_add_table_legend("Iris dataset, column 1 (mean={round(mean(iris[[1]]), 2)}",
                        bookmark="iris_col1") %>%
  body_add_crosstable(crosstable(iris[1])) %>%
  body_add_normal() %>%
  body_add_table_legend("Iris dataset, column 2 (mean={round(mean(iris[[2]]), 2)}",
                        bookmark="iris_col2",
                        name_format=fp_italic, legend_style="Balloon Text") %>%
  body_add_crosstable(crosstable(iris[2])) %>%
  body_add_normal() %>%
  body_add_normal("There is also the figure \@ref(iris_fig)") %>%
  body_add_gg(p) %>%
  body_add_figure_legend("Iris plot", bookmark="iris_fig")
write_and_open(x)
#If asked to update fields, press "Yes". Otherwise press Ctrl+A then F9 twice for the references
#to appear.

```

---

body\_add\_list

*Add a list to an officer document*


---

**Description**

Add a list to an officer document

**Usage**

```
body_add_list(doc, value, ordered = FALSE, style = NULL, ...)
```

```
body_add_list_item(doc, value, ordered = FALSE, style = NULL, ...)
```

**Arguments**

doc	a docx object
value	a character vector (body_add_list()) or scalar (body_add_list_item). See Section below for markdown support.
ordered	if TRUE, adds an ordered list, if FALSE (default), adds a bullet list

style	specify the style manually, overriding ordered. A better way is to set options <code>crosstable_style_list_ordered</code> and <code>crosstable_style_list_unordered</code> globally.
...	passed on to <code>officer::body_add_par()</code>

### Details

Ordered lists and bullet lists are not supported by the default officer template (see <https://github.com/davidgohel/officer/issues>). You have to manually set custom styles matching those list in a custom Word template file. Then, you can use either the `style` argument or `crosstable` options. See examples for more details.

### Value

The docx object `doc`

### Markdown support

In all `crosstable` helpers for `officer`, you can use the following Markdown syntax to format your text:

- *bold*: "`**text in bold**`"
- *italics*: "`*text in italics*`"
- *subscript*: "`Text in ~subscript~`"
- *superscript*: "`Text in ^superscript^`"
- *newline*: Before `<br>` After
- *color*: "`<color:red>red text</color>`"
- *shade*: "`<shade:yellow>yellow text</shade>`" (background color)
- *font family*: "`<ff:symbol>symbol</ff>`" (

Note that the font name depends on your system language. For instant, in French, it would be `Symbol` with an uppercase first letter.

See the last example of `body_add_normal()` for a practical case.

### Author(s)

Dan Chaltiel

### Examples

```
## Not run:
#For this example to work, `my_template.docx` should include styles named
#`ordered_list` and `unordered_list`
```

```
library(officer)
library(crosstable)
options(crosstable_style_list_ordered="ordered_list")
options(crosstable_style_list_unordered="unordered_list")
```

```

read_docx("my_template.docx") %>%
  body_add_list(c("Numbered item 1", "Numbered item 2"), ordered = TRUE) %>%
  body_add_list_item("Numbered item 3", ordered = TRUE) %>%
  body_add_list(c("Bullet item 1", "Bullet item 2"), ordered = FALSE) %>%
  body_add_list_item("Bullet item 3", ordered = FALSE) %>%
  write_and_open()

## End(Not run)

```

---

body_add_normal	<i>Add a new paragraph with default style</i>
-----------------	---

---

## Description

Add a new paragraph in an officier document with default style.

Variables can be inserted in the text as multiple strings (`paste()` style) or enclosed by braces (`glue()` style).

Basic markdown syntax is available: **bold**, *italic*, and underlined.

References to any bookmark can be inserted using the syntax `@ref(bookmark)` and newlines can be inserted using the token `<br>`.

## Usage

```

body_add_normal(
  doc,
  ...,
  .sep = "",
  style = NULL,
  squish = TRUE,
  parse = c("ref", "format", "code")
)

```

## Arguments

doc	the doc object (created with the <code>read_docx</code> function of <code>officer</code> package)
...	one or several character strings, pasted using <code>.sep</code> . As with <code>glue::glue()</code> , expressions enclosed by braces will be evaluated as R code. If more than one variable is passed, all should be of length 1.
.sep	Separator used to separate elements.
style	Style for normal text. Best set with <code>crosstable_options()</code> .
squish	Whether to squish the result (remove trailing and repeated spaces). Default to TRUE. Allows to add multiline paragraph without breaking the string.
parse	which format to parse. Default to all formats ( <code>c("ref", "format", "code")</code> ).

## Value

a new doc object

The docx object `doc`

## Markdown support

In all crosstable helpers for officer, you can use the following Markdown syntax to format your text:

- *bold*: `"**text in bold**"`
- *italics*: `"*text in italics*"`
- *subscript*: `"Text in ~subscript~"`
- *superscript*: `"Text in ^superscript^"`
- *newline*: Before `<br>` After
- *color*: `"<color:red>red text</color>"`
- *shade*: `"<shade:yellow>yellow text</shade>"` (background color)
- *font family*: `"<ff:symbol>symbol</ff>"` (

Note that the font name depends on your system language. For instant, in French, it would be Symbol with an uppercase first letter.

See the last example of `body_add_normal()` for a practical case.

## Author(s)

Dan Chaltiel

## Examples

```
library(officer)
library(crosstable)

info_rows = c("Also, table iris has {nrow(iris)} rows.",
              "And table mtcars has {nrow(mtcars)} rows.")
doc = read_docx() %>%
  body_add_normal("Table iris has", ncol(iris), "columns.", .sep=" ") %>% #paste style
  body_add_normal("However, table mtcars has {ncol(mtcars)} columns") %>% #glue style
  body_add_normal(info_rows) %>% #vector style
  body_add_normal("")
doc = doc %>%
  body_add_normal("You can write text in italic1, underlined1, bold1, and `code`,
                  and you can also add references, for instance a ref to Table
                  @ref(my_table). Multiple spaces are ignored (squished) so that you
                  can enter multiline text.") %>%
  body_add_normal() %>%
  body_add_normal("Here I should use `body_add_crosstable()` to add a table before the
                  legend.") %>%
  body_add_table_legend("My pretty table", bookmark="my_table")
write_and_open(doc)

#Markdown support
read_docx() %>%
  body_add_normal("This is bold and italic (see Table @ref(my_bkm)). <br> This is
                  bold console \\*CODE\\* and bold_and_italic") %>%
  body_add_normal("This is <color:red>red bold text</color>, this is ~subscript italic~",
```



```

        and this is and this is ^superscript with <shade:yellow>yellow</shade>^") %>%
body_add_normal("This is <ff:Alibi>a fancy font</ff> and this `is code`!!") %>%
        #you might need to change "Alibi" to "alibi" here
body_add_normal() %>%
body_add_table_legend("Some table legend", bookmark="my_bkm") %>%
write_and_open()

```

---

body\_add\_table\_list    *Add a list of tables*

---

## Description

Add a list of tables in an officer document. crosstables will be added using `body_add_crosstable()` and flextables will be added using `flextable::body_add_flextable()`. Plain dataframes will be converted to flextables.

## Usage

```

body_add_table_list(
  doc,
  l,
  fun_before = "title2",
  fun_after = NULL,
  fun = fun_before,
  ...
)

body_add_flextable_list(...)

body_add_crosstable_list(...)

```

## Arguments

doc	a rdocx object, created by <code>officer::read_docx()</code>
l	a named list of tables (of class <code>crosstable</code> , <code>flextable</code> , or <code>data.frame</code> ).
fun_before	a function to be used before each table
fun_after	a function to be used after each table.
fun	Deprecated
...	arguments passed on to <code>body_add_crosstable()</code> or <code>body_add_flextable()</code>

## Value

The docx object doc

fun\_before **and** fun\_after

These should be function of the form `function(doc, .name)` where `.name` is the name of the current table of the list. You can also pass "title2" to add the name as a title of level 2 between each table (works for levels 3 and 4 as well), "newline" to simply add a new line, or even NULL to not separate them (beware that the tables might merge then). `fun_before` is designed to add a title while `fun_after` is designed to add a table legend (cf. examples).

## Examples

```
library(officer)
ctl = list(iris2=crosstable(iris2, 1),
          "Just a flextable"=flextable::flextable(mtcars2[1:5,1:5]),
          "Just a dataframe"=iris2[1:5,1:5])

fun1 = function(doc, .name){
  doc %>%
    body_add_title(" This is table '{.name}' as a flex/crosstable", level=2) %>%
    body_add_normal("Here is the table:")
}
fun2 = function(doc, .name){
  doc %>% body_add_table_legend("{.name}", bookmark=.name)
}
read_docx() %>%
  body_add_title("Separated by subtitle", 1) %>%
  body_add_table_list(ctl, fun_before="title2") %>%
  body_add_break() %>%
  body_add_title("Separated using a custom function", 1) %>%
  body_add_normal("You can therefore use bookmarks, for instance here are
                 tables \\@ref(iris2), \\@ref(just_a_flextable)
                 and \\@ref(just_a_dataframe).") %>%
  body_add_table_list(ctl, fun_before=fun1, fun_after=fun2, body_fontsize=8) %>%
  write_and_open()
```

---

body\_add\_table\_section

*Add a section with a table and its legend*

---

## Description

Add a section with a table and its legend

## Usage

```
body_add_table_section(
  doc,
  x,
  legend,
  ...,
```

```

bookmark = NULL,
title = getOption("crosstable_section_title", TRUE),
title_lvl = getOption("crosstable_section_title_level", 3),
sentence = getOption("crosstable_section_sentence", FALSE)
)

```

### Arguments

doc	a rdocx object
x	a table: crosstable, flextable, or plain old dataframe
legend	the legend to use
...	passed on to <a href="#">body_add_flextable()</a> or <a href="#">body_add_crosstable()</a>
bookmark	the bookmark to use. Defaults to the cleaned variable name of x
title	the title to add for the section. Can also be FALSE (no title) or TRUE (the title defaults to legend)
title_lvl	the title level if applicable
sentence	a sentence to add between the title (if applicable) and the table. If TRUE, defaults to "Information about {tolower(title)} is described in Table @ref({bookmark})".

### Value

The docx object doc

### Examples

```

library(officer)
read_docx() %>%
  body_add_title("Description", 1) %>%
  body_add_title("Population A", 2) %>%
  body_add_table_section(head(iris), "The iris dataset", sentence=TRUE) %>%
  body_add_table_section(crosstable(iris), "A crosstable of the iris dataset",
                          title=FALSE, sentence=TRUE, body_fontsize=8) %>%
write_and_open()

```

---

body_add_title	<i>Add a title to an officer document</i>
----------------	---

---

### Description

Add a title to an officer document

**Usage**

```
body_add_title(
  doc,
  value,
  level = 1,
  squish = TRUE,
  style = getOption("crosstable_style_heading", "heading")
)
```

**Arguments**

doc	the doc object (created with the read_docx function of officer package)
value	a character string. See Section below for markdown support.
level	the level of the title. See styles_info(doc) to know the possibilities.
squish	Whether to squish the result (remove trailing and repeated spaces). Default to TRUE.
style	the name of the title style. See styles_info(doc) to know the possibilities.

**Value**

The docx object doc

**Markdown support**

In all crosstable helpers for officer, you can use the following Markdown syntax to format your text:

- *bold*: `"**text in bold**"`
- *italics*: `"*text in italics*"`
- *subscript*: `"Text in ~subscript~"`
- *superscript*: `"Text in ^superscript^"`
- *newline*: Before `<br>` After
- *color*: `"<color:red>red text</color>"`
- *shade*: `"<shade:yellow>yellow text</shade>"` (background color)
- *font family*: `"<ff:symbol>symbol</ff>"` (

Note that the font name depends on your system language. For instant, in French, it would be Symbol with an uppercase first letter.

See the last example of `body_add_normal()` for a practical case.

**Author(s)**

Dan Chaltiel

**Examples**

```
library(officer)
library(crosstable)
library(dplyr)
doc = read_docx() %>%
  body_add_title("La table iris (nrow={nrow(iris)})", 1) %>%
  body_add_title("Description", 2) %>%
  body_add_normal("La table iris a ", ncol(iris), " colonnes.")
#write_and_open(doc)
```

---

body\_replace\_text\_at\_bkms

*Replace text on several bookmarks at once*

---

**Description**

Replace text on several bookmarks at once

**Usage**

```
body_replace_text_at_bkms(doc, ...)
```

**Arguments**

doc	a rdocx object
...	named

**Value**

The docx object doc

**Author(s)**

Dan Chaltiel

---

clean\_names\_with\_labels

*Cleans names of a dataframe while retaining old names as labels*

---

**Description**

Cleans names of a dataframe while retaining old names as labels

**Usage**

```
clean_names_with_labels(
  df,
  except = NULL,
  .fun = getOption("crosstable_clean_names_fun")
)
```

**Arguments**

df	a data.frame
except	<tidy-select> columns that should not be renamed.
.fun	the function used to clean the names. Default function is limited; if the cleaning is not good enough you could use <code>janitor::make_clean_names()</code>

**Value**

A dataframe with clean names and label attributes

**Author(s)**

Dan Chaltiel

**Examples**

```
#options(crosstable_clean_names_fun=janitor::make_clean_names)
x = data.frame("name with space"=1, TwoWords=1, "total $ (2009)"=1, accents=1,
              check.names=FALSE)
cleaned = clean_names_with_labels(x, except=TwoWords)
cleaned %>% names()
cleaned %>% get_label()
```

---

confint_numeric	<i>Confidence interval of a numeric vector</i>
-----------------	--

---

**Description**

Not an S3 method, which might have conflicted with `stats::confint`.

**Usage**

```
confint_numeric(object, level = 0.95, B = 0)
```

**Arguments**

object	a vector, numeric or equivalent (date, logical...)
level	the confidence level required
B	if >0, the number of bootstraps

**Value**

the vector [conf\_inf, conf\_sup]

**Author(s)**

Dan Chaltiel

**Examples**

```
confint_numeric(iris$Sepal.Length)
confint_numeric(mtcars2$hp_date)
confint_numeric(mtcars2$hp_date, level=0.99)
```

---

crosstable

*Easily describe datasets*

---

**Description**

Generate a descriptive table of all chosen columns, as contingency tables for categorical variables and as calculation summaries for numeric variables. If the `by` argument points to one or several categorical variables, `crosstable` will output a description of all columns for each level. Otherwise, if it points to a numeric variable, `crosstable` will calculate correlation coefficients with all other selected numeric columns. Finally, if it points to a `Surv` object, `crosstable` will describe the survival at different times.

Can be formatted as an HTML table using `as_flextable()`.

**Usage**

```
crosstable(
  data,
  cols = everything(),
  ...,
  by = NULL,
  total = c("none", "row", "column", "both"),
  percent_pattern = "{n} ({p_row})",
  percent_digits = 2,
  num_digits = 1,
  showNA = c("ifany", "always", "no"),
  label = TRUE,
  funs = c(` ` = cross_summary),
  funs_arg = list(),
  cor_method = c("pearson", "kendall", "spearman"),
  drop_levels = FALSE,
  unique_numeric = 3,
  date_format = NULL,
  times = NULL,
```

```

followup = FALSE,
test = FALSE,
test_args = crosstable_test_args(),
effect = FALSE,
effect_args = crosstable_effect_args(),
margin = deprecated(),
.vars = deprecated()
)

```

## Arguments

data	A data.frame
cols	<tidy-select> Columns to describe, default to everything(). See examples or vignette("crosstable-selection") for more details.
...	Unused. All parameters after this one must be named.
by	The variable to group on. Character or name.
total	one of ["none", "row", "column" or "both"] to indicate whether to add total rows and/or columns. Default to none.
percent_pattern	Pattern used to describe proportions in categorical data. Syntax uses a <a href="#">glue::glue()</a> specification, see the <b>section</b> below for more details. Default to "{n} ({p_col})" if by is null and "{n} ({p_row})" if it is not.
percent_digits	Number of digits for percentages.
num_digits	Number of digits for numeric summaries.
showNA	Whether to show NA in categorical variables (one of c("ifany", "always", "no"), like in table()).
label	Whether to show labels. See <a href="#">import_labels()</a> or <a href="#">set_label()</a> for how to add labels to the dataset columns.
funs	Functions to apply to numeric variables. Default to <a href="#">cross_summary()</a> .
funs_arg	Additional parameters for funs, e.g. digits (the number of decimal places) for the default <a href="#">cross_summary()</a> . Ultimately, these arguments are passed to <a href="#">format_fixed()</a> .
cor_method	One of c("pearson", "kendall", "spearman") to indicate which correlation coefficient is to be used.
drop_levels	Whether to drop unused levels of factor variables. Default to TRUE.
unique_numeric	The number of non-missing different levels a variable should have to be considered as numeric.
date_format	if x is a vector of Date or POSIXt, the format to apply (see <a href="#">strptime</a> for formats)
times	When using formula with <a href="#">survival::Surv()</a> objects, which times to summarize.
followup	When using formula with <a href="#">survival::Surv()</a> objects, whether to display follow-up time.
test	Whether to perform tests.



test_args	See <a href="#">crosstable_test_args</a> to override default testing behaviour.
effect	Whether to compute a effect measure.
effect_args	See <a href="#">crosstable_effect_args</a> to override default behaviour.
margin	Deprecated in favor of percent_pattern. One of ["row", "column", "cell", "none", or "all"]. Default to row.
.vars	Deprecated in favor of cols.

## Value

A data.frame/tibble of class crosstable

## About percent\_pattern

The percent\_pattern argument is very powerful but can be difficult to understand at first :

- It is usually a single string that uses the glue syntax, where variables are put in curly braces ({x}).
- Counts are expressed as {n}, {n\_row}, {n\_col}, and {n\_tot}, and proportions as {p\_row}, {p\_col}, and {p\_cell}, depending on the margin on which they are calculated.
- For each variable, a version including missing values in the total is proposed as {n\_xxx\_na} or {p\_xxx\_na}.
- For each proportion, a confidence interval is also calculated using **Wilson score** and can be expressed as {p\_xxx\_inf} and {p\_xxx\_sup}. See examples for practical applications.
- Alternatively, percent\_pattern can be a list of characters with names body, total\_row, total\_col, and total\_all to also control the pattern in other parts of the crosstable than the body.

## Author(s)

Dan Chaltiel

## See Also

<https://danchaltiel.github.io/crosstable/>, [as\\_flextable](#), [import\\_labels](#)

## Examples

```
#whole table
crosstable(iris)
crosstable(mtcars)
crosstable(mtcars2)

#tidyselection, custom functions
library(dplyr)
crosstable(mtcars2, c(ends_with("t"), starts_with("c")), by=vs,
           funs=c(mean, quantile), funs_arg=list(probs=c(.25,.75)))

#margin and totals, multiple by
```

```

crosstable(mtcars2, c(displ, cyl), by=c(am, vs),
           margin=c("row", "col"), total = "both")

#predicate selection, correlation, effect calculation
crosstable(mtcars2, where(is.numeric), by=hp, effect=TRUE)

#lambda selection & statistical tests
crosstable(mtcars2, ~is.numeric(.x) && mean(.x)>50, by=vs, test=TRUE)

#Dates
mtcars2$my_date = as.Date(mtcars2$hp , origin="2010-01-01") %>% set_label("Some nonsense date")
crosstable(mtcars2, my_date, by=vs, date_format="%d/%m/%Y")

#Survival data (using formula syntax)
library(survival)
crosstable(aml, Surv(time, status) ~ x, times=c(0,15,30,150), followup=TRUE)

#Patterns
crosstable(mtcars2, vs, by=am, percent_digits=0,
           percent_pattern="{n} ({{p_col}} / {{p_row}})")
crosstable(mtcars2, vs, by=am, percent_digits=0,
           percent_pattern="N={n} \np[95%CI] = {{p_col}} [{{p_col_inf}}; {{p_col_sup}}]")
str_high="n>5"; str_lo="n<=5"
crosstable(mtcars2, vs, by=am, percent_digits=0,
           percent_pattern="col={{p_col}}, row={{p_row}} ({{ifelse(n<5, str_lo, str_high}})")

```

---

crosstable\_effect\_args

*Default arguments for calculating and displaying effects in `crosstable()`*

---

## Description

This helper function provides default parameters for defining how the effect sizes should be computed. It belongs to the `effect_args` argument of the `crosstable()` function. See [effect\\_summary](#), [effect\\_tabular](#), and [effect\\_survival](#) for more insight.

## Usage

```

crosstable_effect_args(
  effect_summarize = diff_mean_auto,
  effect_tabular = effect_odds_ratio,
  effect_survival = effect_survival_coxph,
  effect_display = display_effect,
  conf_level = 0.95,
  digits = 2
)

```

**Arguments**

<code>effect_summarize</code>	a function of three arguments (continuous variable, grouping variable and <code>conf_level</code> ), used to compare continuous variable. Returns a list of five components: <code>effect</code> (the effect value(s)), <code>ci</code> (the matrix of confidence interval(s)), <code>effect.name</code> (the interpretation(s) of the effect value(s)), <code>effect.type</code> (the description of the measure used) and <code>conf_level</code> (the confidence interval level). Users can use <a href="#">diff_mean_auto()</a> , <a href="#">diff_mean_student()</a> , <a href="#">diff_mean_boot()</a> , or <a href="#">diff_median()</a> , or their custom own function.
<code>effect_tabular</code>	a function of three arguments (two categorical variables and <code>conf_level</code> ) used to measure the associations between two factors. Returns a list of five components: <code>effect</code> (the effect value(s)), <code>ci</code> (the matrix of confidence interval(s)), <code>effect.name</code> (the interpretation(s) of the effect value(s)), <code>effect.type</code> (the description of the measure used) and <code>conf_level</code> (the confidence interval level). Users can use <a href="#">effect_odds_ratio()</a> , <a href="#">effect_relative_risk()</a> , or <a href="#">effect_risk_difference()</a> , or their custom own function.
<code>effect_survival</code>	a function of two argument (a formula and <code>conf_level</code> ), used to measure the association between a censored and a factor. Returns the same components as created by <code>effect_summarize</code> . Users can use <a href="#">effect_survival_coxph()</a> or their custom own function.
<code>effect_display</code>	a function to format the effect. See <a href="#">display_effect()</a> .
<code>conf_level</code>	the desired confidence interval level
<code>digits</code>	the decimal places

**Value**

A list with effect parameters

**Author(s)**

Dan Chaltiel

---

`crosstable_options`      *Options for the package crosstable*

---

**Description**

Use this function to manage your `crosstable` parameters globally while taking advantage of auto-completion. Use [crosstable\\_peek\\_options\(\)](#) to see which option is currently set and [crosstable\\_reset\\_options\(\)](#) to set all options back to default.

**Usage**

```
crosstable_options(  
  ...,  
  zero_percent = FALSE,  
  only_round = FALSE,  
  verbosity_autotesting = "default",  
  verbosity_duplicate_cols = "default",  
  crosstable_fishtest_B = 1e+05,  
  total,  
  percent_pattern,  
  margin,  
  percent_digits,  
  num_digits,  
  showNA,  
  label,  
  funs,  
  funs_arg,  
  cor_method,  
  drop_levels,  
  unique_numeric,  
  date_format,  
  times,  
  followup,  
  test_arg,  
  effect_args,  
  wrap_id = 70,  
  compact_padding = 25,  
  header_show_n_pattern = "{.col} (N={.n})",  
  keep_id,  
  by_header,  
  autofit,  
  compact,  
  remove_header_keys,  
  show_test_name,  
  padding_v,  
  header_show_n,  
  fontsize_body,  
  fontsize_subheaders,  
  fontsize_header,  
  units = "in",  
  peek_docx = TRUE,  
  font_code = "Consolas",  
  add_max_cols = 25,  
  format_legend_name,  
  table_legend_par_before,  
  table_legend_prefix,  
  figure_legend_par_after,  
  figure_legend_prefix,  
)
```

```

    normal_squish,
    title_squish,
    allow_break,
    style_normal,
    style_character,
    style_strong,
    style_image,
    style_legend,
    style_heading,
    style_list_ordered,
    style_list_unordered,
    scientific_log,
    .local = FALSE,
    reset = deprecated()
)

```

### Arguments

...	unused
zero_percent	set to TRUE so that proportions are not displayed if n==0
only_round	default argument for <a href="#">format_fixed()</a>
verbosity_autotesting	one of default, quiet, or verbose
verbosity_duplicate_cols	one of default, quiet, or verbose.
crosstable_fishertest_B	number of simulations to perform when <code>fisher.test()</code> is failing (FEXACT error 7).
total	For setting <a href="#">crosstable()</a> arguments globally.
percent_pattern	For setting <a href="#">crosstable()</a> arguments globally.
margin	For setting <a href="#">crosstable()</a> arguments globally.
percent_digits	For setting <a href="#">crosstable()</a> arguments globally.
num_digits	For setting <a href="#">crosstable()</a> arguments globally.
showNA	For setting <a href="#">crosstable()</a> arguments globally.
label	For setting <a href="#">crosstable()</a> arguments globally.
funs	For setting <a href="#">crosstable()</a> arguments globally.
funs_arg	For setting <a href="#">crosstable()</a> arguments globally.
cor_method	For setting <a href="#">crosstable()</a> arguments globally.
drop_levels	For setting <a href="#">crosstable()</a> arguments globally.
unique_numeric	For setting <a href="#">crosstable()</a> arguments globally.
date_format	For setting <a href="#">crosstable()</a> arguments globally.
times	For setting <a href="#">crosstable()</a> arguments globally.

followup	For setting <code>crosstable()</code> arguments globally.
test_arg	For setting <code>crosstable()</code> arguments globally.
effect_args	For setting <code>crosstable()</code> arguments globally.
wrap_id	if id contains no spaces, wrap it with this maximum number of characters.
compact_padding	in flextables, left-padding for non-headers rows when compact=TRUE.
header_show_n_pattern	glue pattern used when showing N in the header of flextables. <code>.col</code> is the name of the column and <code>.n</code> the size of the group. Default to <code>{.col}</code> ( <code>N={.n}</code> ).
keep_id	For setting <code>as_flextable()</code> arguments globally.
by_header	For setting <code>as_flextable()</code> arguments globally.
autofit	For setting <code>as_flextable()</code> arguments globally.
compact	For setting <code>as_flextable()</code> arguments globally.
remove_header_keys	For setting <code>as_flextable()</code> arguments globally.
show_test_name	For setting <code>as_flextable()</code> arguments globally.
padding_v	For setting <code>as_flextable()</code> arguments globally.
header_show_n	For setting <code>as_flextable()</code> arguments globally.
fontsize_body	For setting <code>as_flextable()</code> arguments globally.
fontsize_subheaders	For setting <code>as_flextable()</code> arguments globally. Subheaders are only considered when compact=TRUE.
fontsize_header	For setting <code>as_flextable()</code> arguments globally.
units	default units in <code>body_add_gg2()</code> and <code>body_add_img2()</code>
peek_docx	behavior of <code>peek()</code> , which will open a docx if TRUE (default) and an xlsx if FALSE
font_code	font family used to show code, most likely a monospaced typeface such as Consolas (default)
add_max_cols	max number of columns a crosstable can have to be added to a Word document
format_legend_name	how the legend name ("Table", "Figure") is formatted. Default to <code>officer::fp_text_lite(bold=TRUE)</code>
table_legend_par_before	whether to add an empty paragraph before all table legends
table_legend_prefix, figure_legend_prefix	a prefix before each legend, after the numbering
figure_legend_par_after	whether to add an empty paragraph after all figure legends
normal_squish	Should you squish text in normal paragraphs?
title_squish	Should you squish text in headers paragraphs?
allow_break	allow crosstable rows to break across pages

style_normal	For specifying styles used in your {officer} template.
style_character	For specifying styles used in your {officer} template.
style_strong	For specifying styles used in your {officer} template.
style_image	For specifying styles used in your {officer} template.
style_legend	For specifying styles used in your {officer} template.
style_heading	For specifying styles used by headings on different levels. Levels will be pasted in the end (e.g. use "title" if your level 2 heading style is "title2").
style_list_ordered, style_list_unordered	For specifying styles used by lists in the rdocx template. Needed for <code>body_add_list()</code> to work.
scientific_log	the maximum power a number can have before being formatted as scientific. Default to 4 so applies on numbers $<1e-4$ or $>1e4$ .
.local	if TRUE, the effect will only apply to the local frame (thanks to <code>rlang::local_options()</code> )
reset	if TRUE, set all these options back to default

**Value**

Nothing, called for its side effects

**See Also**

[crosstable\\_peek\\_options\(\)](#) and [crosstable\\_reset\\_options\(\)](#)

---

crosstable\_peek\_options

*See which crosstable option is currently set.*

---

**Description**

See which crosstable option is currently set.

**Usage**

```
crosstable_peek_options(keep_null = FALSE)
```

**Arguments**

keep\_null        set to TRUE to get a list

**Value**

A named list of crosstable options

---

`crosstable_reset_options`*Reset all crosstable options.*

---

**Description**

Reset all crosstable options.

**Usage**

```
crosstable_reset_options(quiet = FALSE)
```

**Arguments**

`quiet` set to TRUE to remove the message.

**Value**

Nothing, called for its side effects

---

`crosstable_test_args` *Default arguments for calculating and displaying tests in [crosstable\(\)](#)*

---

**Description**

This is the starting point for refining the testing algorithm used in `crosstable`. Users can provide their own functions for `test.~`.

**Usage**

```
crosstable_test_args(  
  test_summarize = test_summarize_auto,  
  test_tabular = test_tabular_auto,  
  test_correlation = test_correlation_auto,  
  test_survival = test_survival_logrank,  
  test_display = display_test,  
  plim = 4,  
  show_method = TRUE  
)
```



**Arguments**

<code>test_summarize</code>	a function of two arguments (continuous variable and grouping variable), used to compare continuous variable. Must return a list of two components: <code>p.value</code> and <code>method</code> . See <a href="#">test_summarize_auto</a> or <a href="#">test_summarize_linear_contrasts</a> for some examples of such functions.
<code>test_tabular</code>	a function of two arguments (two categorical variables), used to test association between two categorical variables. Must return a list of two components: <code>p.value</code> and <code>method</code> . See <a href="#">test_tabular_auto</a> for example.
<code>test_correlation</code>	a function of three arguments (two continuous variables plus the correlation method), used to test association between two continuous variables. Like <code>cor.test</code> , it must return a list of at least <code>estimate</code> , <code>p.value</code> , and <code>method</code> , with also <code>conf.int</code> optionally. See <a href="#">test_correlation_auto</a> for example.
<code>test_survival</code>	a function of one argument (the formula <code>surv~by</code> ), used to compare survival estimations. Must return a list of two components: <code>p.value</code> and <code>method</code> . See <a href="#">test_survival_logrank</a> for example.
<code>test_display</code>	function used to display the test result. See <a href="#">display_test</a> .
<code>plim</code>	number of digits for the p value.
<code>show_method</code>	whether to display the test name (logical).

**Value**

A list with test parameters

**Author(s)**

Dan Chaitiel

**See Also**

[test\\_summarize\\_auto](#), [test\\_tabular\\_auto](#), [test\\_survival\\_logrank](#), [test\\_summarize\\_linear\\_contrasts](#), [display\\_test](#)

**Examples**

```
library(dplyr)
my_test_args=crosstable_test_args()
my_test_args$test_summarize = test_summarize_linear_contrasts
iris %>%
  mutate(Petal.Width.qt = paste0("Q", ntile(Petal.Width, 5)) %>% ordered()) %>%
  crosstable(Petal.Length ~ Petal.Width.qt, test=TRUE, test_args = my_test_args)
```

---

cross_summary	<i>Summarize a numeric vector</i>
---------------	-----------------------------------

---

**Description**

Summarize a numeric vector with min, max, mean, sd, median, IQR, n and missings.

**Usage**

```
cross_summary(x, dig = 1, ...)
```

**Arguments**

x	a numeric vector
dig	number of digits
...	params to pass on to <code>format_fixed()</code> : <code>zero_digits</code> and <code>only_round</code>

**Value**

a list of named functions

**Author(s)**

Dan Chaltiel, David Hajage

**Examples**

```
cross_summary(iris$Sepal.Length)
cross_summary(iris$Petal.Width, dig=3)
cross_summary(mtcars2$hp_date)
cross_summary(mtcars2$qsec_posix, date_format="%d/%m %H:%M")
```

---

ct_compact	<i>Generic function to compact a table (publication formatting)</i>
------------	---

---

**Description**

Generic function to compact a table (publication formatting)

**Usage**

```
## S3 method for class 'data.frame'
ct_compact(
  data,
  name_from,
  name_to = "variable",
  wrap_cols = NULL,
  rtn_flextable = FALSE,
  ...
)

## S3 method for class 'crosstable'
ct_compact(
  data,
  name_from = c("label", ".id"),
  name_to = "variable",
  keep_id = FALSE,
  ...
)
```

**Arguments**

data	the object to compact
...	additional arguments (not used)
name_from	name of the column to be collapsed when compacting
name_to	name of the column that will receive the collapsed column. Will be created if it doesn't exist.
wrap_cols	name of the columns to wrap
rtn_flextable	whether to return a formatted <code>flextable()</code> object or a simple <code>data.frame</code>
keep_id	glue pattern to keep the column name along with the label. If TRUE, default to "{label} ({.id})".

**Value**

a compacted `data.frame`

**Author(s)**

Dan Chaitiel

**Examples**

```
#dataframes
x=iris[c(1:5,51:55,101:105),]
ct_compact(x, name_from="Species")
ct_compact(x, name_from="Species", name_to="Petal.Length")
x$Species2 = substr(x$Species, 1, 1)
```

```

ct_compact(x, name_from="Species", wrap_cols="Species2")

#crosstables
x=crosstable(mtcars2, c(displ, hp, am), by=vs, test=TRUE, effect=TRUE)
ct_compact(x)
ct_compact(x, name_from=".id")

```

---

display\_effect            *Default function to display the effect*

---

### Description

User can provide their own custom version in [crosstable\\_effect\\_args\(\)](#)

### Usage

```
display_effect(effect, digits = 4)
```

### Arguments

effect	effect
digits	digits

### Value

a character vector

### Author(s)

Dan Chaitiel

---

display\_test            *Default function to display a test result*

---

### Description

Default function to display a test result

### Usage

```
display_test(test, digits = 4, method = TRUE)
```

### Arguments

test	test
digits	number of digits
method	display method

**Value**

a string

**Author(s)**

Dan Chaltiel

---

docx_bookmarks2	<i>List Word bookmarks, including the ones in header and footer</i>
-----------------	---

---

**Description**

This is a correction of `officer::docx_bookmarks()`. See [this PR](#).

**Usage**

```
docx_bookmarks2(  
  x,  
  return_vector = FALSE,  
  target = c("all", "header", "body", "footer")  
)
```

**Arguments**

x	an rdocx object
return_vector	use TRUE for compatibility with <code>officer::docx_bookmarks()</code>
target	one of <code>c("all", "header", "body", "footer")</code>

**Value**

a list with all bookmarks

**Author(s)**

Dan Chaltiel

---

effect_summary	<i>Effect measure for association between one continuous and one categorical variable</i>
----------------	---

---

### Description

User can either use or extend these functions to configure effect calculation.

### Usage

```
diff_mean_auto(x, by, conf_level = 0.95, R = 500)
```

```
diff_mean_boot(x, by, conf_level = 0.95, R = 500)
```

```
diff_median_boot(x, by, conf_level = 0.95, R = 500)
```

```
diff_mean_student(x, by, conf_level = 0.95)
```

### Arguments

x	numeric vector
by	categorical vector (of exactly 2 unique levels)
conf_level	confidence interval level
R	number of bootstrap replication

### Value

A list with five components: effect, ci, effect.name, effect.type, and conf\_level

### Functions

- `diff_mean_auto()`: (**Default**) calculate a specific "difference in means" effect based on normality (Shapiro or Anderson test) and variance homogeneity (Bartlett test)
- `diff_mean_boot()`: calculate a "difference in means" effect with a bootstrapped CI using standard deviation
- `diff_median_boot()`: calculate a "difference in medians" effect with a bootstrapped CI using quantiles#'
- `diff_mean_student()`: calculate a "difference in means" effect using t.test confidence intervals

### Author(s)

Dan Chaltiel, David Hajage

### See Also

[crosstable\\_effect\\_args\(\)](#)

---

effect_survival	<i>Effect measure for association between one censored variable and one categorical variable</i>
-----------------	--

---

**Description**

Effect measure for association between one censored variable and one categorical variable

**Usage**

```
effect_survival_coxph(x, by, conf_level = 0.95)
```

**Arguments**

x	survival vector (made using <code>survival::Surv()</code> )
by	categorical vector (of exactly 2 unique levels)
conf_level	confidence interval level

**Value**

a list with two components: p.value and method

**Author(s)**

Dan Chaltiel, David Hajage

---

effect_tabular	<i>Effect measure for association between two categorical variables</i>
----------------	---

---

**Description**

User can either use or extend these functions to configure effect calculation.

**Usage**

```
effect_odds_ratio(x, by, conf_level = 0.95)
effect_relative_risk(x, by, conf_level = 0.95)
effect_risk_difference(x, by, conf_level = 0.95)
```

**Arguments**

x	categorical vector (character, factor, ...)
by	categorical vector (of exactly 2 unique levels)
conf_level	confidence interval level

**Value**

A list with five components: effect, ci, effect.name, effect.type, and conf\_level

**Functions**

- effect\_odds\_ratio(): **(Default)** calculate the odds ratio
- effect\_relative\_risk(): calculate the relative risk
- effect\_risk\_difference(): calculate the risk difference

**Author(s)**

Dan Chaltiel, David Hajage

**See Also**

[crosstable\\_effect\\_args\(\)](#)

---

format_fixed	<i>Format numbers with the exact same number of decimals, including trailing zeros</i>
--------------	--

---

**Description**

Format numbers with the exact same number of decimals, including trailing zeros

**Usage**

```
format_fixed(
  x,
  digits = 1,
  zero_digits = 1,
  date_format = NULL,
  percent = FALSE,
  is_period = FALSE,
  scientific = getOption("crosstable_scientific_log", 4),
  epsilon = getOption("crosstable_format_epsilon", NULL),
  only_round = getOption("crosstable_only_round", FALSE),
  ...
)
```

**Arguments**

x	a numeric vector to format
digits	number of decimals
zero_digits	number of significant digits for values rounded to 0 (can be set to NULL to keep the original 0 value)



date_format	if x is a vector of Date or POSIXt, the format to apply (see <a href="#">strptime</a> for formats)
percent	if TRUE, format the values as percentages
is_period	whether x is a period (a numeric value of seconds)
scientific	the power of ten above/under which numbers will be displayed as scientific notation.
epsilon	values less than epsilon are formatted as "< [epsilon]"
only_round	if TRUE, format_fixed simply returns the rounded value. Can be set globally with options("crosstable_only_round"=TRUE).
...	unused

**Value**

a character vector of formatted numbers

**Author(s)**

Dan Chaitiel

**Examples**

```
x = c(1, 1.2, 12.78749, pi, 0.00000012)
format_fixed(x, digits=3) #default zero_digits=1
format_fixed(x, digits=3, zero_digits=2)
format_fixed(x, digits=3, zero_digits=NULL)

x_sd = sd(iris$Sepal.Length/10000, na.rm=TRUE)
format_fixed(x_sd, dig=6)
format_fixed(x_sd, dig=3, zero_digits=2) #default only_round=FALSE
format_fixed(x_sd, dig=3, zero_digits=2, only_round=TRUE)
options("crosstable_only_round"=TRUE)
format_fixed(x_sd, dig=3, zero_digits=2) #override default
options("crosstable_only_round"=NULL)

x2 = c(0.01, 0.1001, 0.500005, 0.00000012)
format_fixed(x2, scientific=0, dig=1) #everything abs>10^0 gets scientific
#last would be 0 so it is scientific. Try `zero_digits=NA` or `dig=7`
format_fixed(x2, scientific=FALSE, dig=6)
format_fixed(x2, scientific=FALSE, percent=TRUE, dig=0)
format_fixed(x2, scientific=FALSE, eps=0.05)
```

---

generate\_autofit\_macro

*Generate a macro file for autofitting*

---

**Description**

This function generates a file that can be imported into MS Word in order to use a macro for autofitting all tables in a document at once. This macro file should be imported only once per computer.

**Usage**

```
generate_autofit_macro()
```

**Value**

Nothing, called for its side effects

**Installation**

- In the R console, run `generate_autofit_macro()` to generate the file `crosstable_autofit.bas` in your working directory.
- In MS Word, press `Alt+F11` to open the VB Editor.
- In the Editor, go to `File > Import` or press `Ctrl+M` to open the import dialog, and import `crosstable_autofit.bas`. There should now be a "CrosstableMacros" module in the "Normal" project.
- Run the macro, either from the VB Editor or from `View > Macros > View Macros > Run`.

This process will make the macro accessible from any Word file on this computer. Note that, in the Editor, you can also drag the module to your document project to make the macro accessible only from this file. The file will have to be named with the `docm` extension though.

**Author(s)**

Dan Chaltiel

---

```
get_label
```

*Get label if wanted and available, or default (name) otherwise*

---

**Description**

Get label if wanted and available, or default (name) otherwise

**Usage**

```
get_label(x, default = names(x), object = FALSE, simplify = TRUE)
```

**Arguments**

<code>x</code>	labelled object. If <code>x</code> is a list/data.frame, <code>get_label()</code> will return the labels of all children recursively
<code>default</code>	value returned if there is no label. Default to <code>names(x)</code> .
<code>object</code>	if <code>x</code> is a list/data.frame, <code>object=TRUE</code> will force getting the labels of the object instead of the children
<code>simplify</code>	if <code>x</code> is a list and <code>object=FALSE</code> , simplify the result to a vector

**Value**

A character vector if `simplify==TRUE`, a list otherwise

**Author(s)**

Dan Chaltiel

**See Also**

[set\\_label\(\)](#), [import\\_labels\(\)](#), [remove\\_label\(\)](#), [Hmisc::label\(\)](#), [expss::var\\_lab\(\)](#)

**Examples**

```
xx=mtcars2 %>%
  set_label("The mtcars2 dataset", object=TRUE)
xx$cyl=remove_label(xx$cyl)

#vectors
get_label(xx$mpg) #label="Miles/(US) gallon"
get_label(xx$cyl) #default to NULL (as names(xx$cyl)==NULL)
get_label(xx$cyl, default="Default value")

#data.frames
get_label(xx)
get_label(xx, object=TRUE)
data.frame(name=names(xx), label=get_label(xx, default=NA)) #cyl is NA

#lists
get_label(list(xx$cyl, xx$mpg))
get_label(list(foo=xx$cyl, bar=xx$mpg))
get_label(list(foo=xx$cyl, bar=xx$mpg), default="Default value")
```

---

get\_percent\_pattern    *Percent pattern helper*

---

**Description**

Get a list with pre-filled values for `percent_pattern`.

**Usage**

```
get_percent_pattern(
  margin = c("row", "column", "cell", "none", "all"),
  na = FALSE
)
```

**Arguments**

`margin`            a vector giving the margins to compute.  
`na`                    whether to use NA

**Value**

a list

**Examples**

```
get_percent_pattern(c("cells", "row", "column"))
get_percent_pattern(c("cells", "row", "column"), na=TRUE)
```

---

import\_labels

*Import labels*


---

**Description**

import\_labels imports labels from a data.frame (data\_label) to another one (.tbl). Works in synergy with [save\\_labels\(\)](#).

save\_labels saves the labels from a data.frame in a temporary variable that can be retrieve by import\_labels.

**Usage**

```
import_labels(
  .tbl,
  data_label,
  name_from = "name",
  label_from = "label",
  warn_name = FALSE,
  warn_label = FALSE,
  verbose = deprecated()
)

save_labels(.tbl)
```

**Arguments**

.tbl	the data.frame to be labelled
data_label	a data.frame from which to import labels. If missing, the function will take the labels from the last dataframe on which <a href="#">save_labels()</a> was called.
name_from	in data_label, which column to get the variable name (default to name)
label_from	in data_label, which column to get the variable label (default to label)
warn_name	if TRUE, displays a warning if a variable name is not found in data_label
warn_label	if TRUE, displays a warning if a label is not found in .tbl
verbose	deprecated

**Value**

A dataframe, as `.tbl`, with labels  
`.tbl` invisibly. Used only for its side effects.

**Author(s)**

Dan Chaitiel

**See Also**

[get\\_label\(\)](#), [set\\_label\(\)](#), [remove\\_label\(\)](#), [save\\_labels\(\)](#)

**Examples**

```
#import the labels from a data.frame to another
iris_label = data.frame(
  name=c("Sepal.Length", "Sepal.Width",
        "Petal.Length", "Petal.Width", "Species"),
  label=c("Length of Sepals", "Width of Sepals",
         "Length of Petals", "Width of Petals", "Specie name")
)
iris %>%
  import_labels(iris_label) %>%
  crosstable

#save the labels, use some dplyr label-removing function, then retrieve the labels
library(dplyr)
mtcars2 %>%
  save_labels() %>%
  transmute(dispatch=as.numeric(dispatch)+1) %>%
  import_labels(warn_label=FALSE) %>% #
  crosstable(dispatch)
```

---

iris2

*Modified iris dataset*

---

**Description**

Modified iris dataset so:

- every column is labelled (using label attribute)
- Species column is considered as factor

See [iris](#) for more informations on the original "Edgar Anderson's Iris Data" dataset.

**Usage**

```
iris2
```

**Format**

A data frame with 150 observations on 5 variables with labels.

**Source**

```
library(dplyr)
iris2 = iris %>%
  expss::apply_labels( #I also could have used [import_labels] or even `labelled::set_variable_labels`
    Species = "Specie",
    Sepal.Length = "Length of Sepal",
    Sepal.Width = "Width of Sepal",
    Petal.Length = "Length of Petal",
    Petal.Width = "Width of Petal"
  ) %>%
  as_tibble()
```

**Examples**

```
library(crosstable)
ct=crosstable(iris2, by=Species)
ct
as_flextable(ct)
```

---

is.crosstable	<i>Test if an object is a crosstable</i>
---------------	--

---

**Description**

Test if an object is a crosstable

**Usage**

```
is.crosstable(x)

is.transposed_crosstable(x)

is.compacted_crosstable(x)

is.multiby_crosstable(x)
```

**Arguments**

x                    An object

**Value**

TRUE if the object inherits from the crosstable class or other subclasses.

---

`mtcars2`*Modified mtcars dataset*

---

## Description

Modified mtcars dataset so:

- every column is labelled (using `label` attribute)
- rownames are a character column named `model`
- `gear` and `cyl` columns are considered as numerical factors
- `vs` and `am` columns are considered as character vector

See [mtcars](#) for more informations on the original "Motor Trend Car Road Tests" dataset.

## Usage

```
mtcars2
```

## Format

A data frame with 32 observations on 11 variables with labels.

## Source

```
library(dplyr)
mtcars2 = mtcars %>%
  mutate(
    model=rownames(mtcars),
    vs=ifelse(vs==0, "vshaped", "straight"),
    am=ifelse(am==0, "auto", "manual"),
    across(c("cyl", "gear"), factor),
    .before=1
  ) %>%
  expss::apply_labels( #I also could have used [import_labels] or even `labelled::set_variable_labels`
    mpg="Miles/(US) gallon",
    cyl="Number of cylinders",
    disp="Displacement (cu.in.)",
    hp="Gross horsepower",
    drat="Rear axle ratio",
    wt="Weight (1000 lbs)",
    qsec="1/4 mile time",
    vs="Engine",
    am="Transmission",
    gear="Number of forward gears",
    carb="Number of carburetors"
  )
```

**Examples**

```
library(crosstable)
ct=crosstable(mtcars2, by=vs)
ct
as_flextable(ct)
```

---

N	<i>Return the number of non NA observations</i>
---	---

---

**Description**

Return the number of non NA observations

**Usage**

```
N(x)
```

**Arguments**

x                    a vector

**Value**

integer, number of non NA observations

**Author(s)**

David Hajage

---

na	<i>Return the number of NA observations</i>
----	---

---

**Description**

Return the number of NA observations

**Usage**

```
na(x)
```

**Arguments**

x                    a vector

**Value**

integer, number of NA observations



**Author(s)**

David Hajage

---

narm	<i>Remove missing values</i>
------	------------------------------

---

**Description**

Remove missing values

**Usage**

narm(x)

**Arguments**

x                    a vector

**Value**

the same vector without missing values

---

peek	<i>Open a crosstable in a temporary document</i>
------	--

---

**Description**

This eases copy-pasting

**Usage**

peek(x, docx = getOption("crosstable\_peek\_docx", TRUE), ...)

**Arguments**

x	a crosstable
docx	if true, peek as a docx, else, peek as xlsx
...	passed on to as_flextable.crosstable() or to as_workbook()

**Value**

Nothing, called for its side effects

**Author(s)**

Dan Chaltiel

---

<code>pivot_crosstable</code>	<i>Pivot a crosstable</i>
-------------------------------	---------------------------

---

**Description**

Pivot a crosstable so the variable column is spread across its values.

**Usage**

```
pivot_crosstable(ct)
```

**Arguments**

<code>ct</code>	a crosstable
-----------------	--------------

**Value**

a tibble of class `pivoted_crosstable`

**Examples**

```
ct = crosstable(mtcars2, c(mpg, drat, wt, qsec))
p_ct = pivot_crosstable(ct)
as_flextable(p_ct)
```

---

<code>plim</code>	<i>Format p values (alternative to <a href="#">format.pval()</a>)</i>
-------------------	---

---

**Description**

Format p values (alternative to [format.pval\(\)](#))

**Usage**

```
plim(p, digits = 4)
```

**Arguments**

<code>p</code>	p values
<code>digits</code>	number of digits

**Value**

formatted p values

**Author(s)**

David Hajage

**See Also**

[format.pval\(\)](https://stackoverflow.com/a/23018806/3888000), <https://stackoverflow.com/a/23018806/3888000>

---

remove_labels	<i>Remove all label attributes.</i>
---------------	-------------------------------------

---

**Description**

Use `remove_labels()` to remove the label from an object or to recursively remove all the labels from a collection of objects (such as a list or a `data.frame`).

This can be useful with functions reacting badly to labelled objects.

**Usage**

```
remove_labels(x)
```

**Arguments**

x                    object to unlabel

**Value**

An object of the same type as `x`, with no labels

**Author(s)**

Dan Chaltiel

**See Also**

[get\\_label](#), [set\\_label](#), [import\\_labels](#), [expss::unlab](#)

**Examples**

```
mtcars2 %>% remove_labels %>% crosstable(mpg) #no label
mtcars2$hp %>% remove_labels %>% get_label() #NULL
```

---

rename\_with\_labels      *Rename every column of a dataframe with its label*

---

### Description

Rename every column of a dataframe with its label

### Usage

```
rename_with_labels(df, except = NULL)
```

### Arguments

df                      a data.frame  
 except                <tidy-select> columns that should not be renamed.

### Value

A dataframe which names are copied from the label attribute

### Author(s)

Dan Chaltiel

### Source

<https://stackoverflow.com/q/75848408/3888000>

### Examples

```
rename_with_labels(mtcars2[,1:5], except=5) %>% names()
rename_with_labels(iris2, except=Sepal.Length) %>% names()
rename_with_labels(iris2, except=starts_with("Pet")) %>% names()
```

---

set\_label                *Set the "label" attribute of an object*

---

### Description

Set the "label" attribute of an object  
 Copy the label from one variable to another

### Usage

```
set_label(x, value, object = FALSE)

copy_label_from(x, from)
```

**Arguments**

x	the variable to label
value	value of the label. If x is a list/data.frame, the labels will all be set recursively. If value is a function, it will be applied to the current labels of x.
object	if x is a list/data.frame, object=TRUE will force setting the labels of the object instead of the children
from	the variable whose label must be copied

**Value**

An object of the same type as x, with labels

**Author(s)**

Dan Chaltiel

**See Also**

[get\\_label\(\)](#), [import\\_labels\(\)](#), [remove\\_label\(\)](#)

**Examples**

```
library(dplyr)
mtcars %>%
  mutate(mpg2=set_label(mpg, "Miles per gallon"),
         mpg3=mpg %>% copy_label_from(mpg2)) %>%
  crosstable(c(mpg, mpg2, mpg3))
mtcars %>%
  copy_label_from(mtcars2) %>%
  crosstable(c(mpg, vs))
mtcars2 %>% set_label(toupper) %>% get_label()
```

---

summaryFunctions

*Summary functions*

---

**Description**

Summary functions to use with [crosstable\(\)](#) or anywhere else.

**Usage**

```
meansd(x, na.rm = TRUE, dig = 2, ...)
```

```
meanCI(x, na.rm = TRUE, dig = 2, level = 0.95, format = TRUE, ...)
```

```
mediqr(x, na.rm = TRUE, dig = 2, format = TRUE, ...)
```

```
minmax(x, na.rm = TRUE, dig = 2, ...)
```

```
nna(x)
```

### Arguments

x	a numeric vector
na.rm	TRUE as default
dig	number of digits
...	params to pass on to <a href="#">format_fixed()</a> : <ul style="list-style-type: none"><li>• <code>zero_digits</code> (default=1): the number of significant digits for values rounded to 0 (set to NULL to keep the original 0 value)</li><li>• <code>only_round</code> (default=FALSE): use <a href="#">round()</a> instead of <a href="#">format_fixed()</a></li></ul>
level	the confidence level required
format	a sugar argument. If FALSE, the function returns a list instead of a formatted string

### Value

a character vector

### Functions

- [meansd\(\)](#): returns mean and std error
- [meanCI\(\)](#): returns mean and confidence interval
- [mediqr\(\)](#): returns median and IQR
- [minmax\(\)](#): returns minimum and maximum
- [nna\(\)](#): returns number of observations and number of missing values

### Fixed format

These functions use [format\\_fixed\(\)](#) which allows to have trailing zeros after rounded values. In the case when the output of rounded values is zero, the use of the `zero_digits` argument allows to keep some significant digits for this specific case only.

### Author(s)

Dan Chaltiel, David Hajage

### See Also

[format\\_fixed\(\)](#)

**Examples**

```

meansd(iris$Sepal.Length, dig=3)
meanCI(iris$Sepal.Length)
minmax(iris$Sepal.Length, dig=3)
mediqr(iris$Sepal.Length, dig=3)
nna(iris$Sepal.Length)

#arguments for format_fixed
x = iris$Sepal.Length/10000 #closer to zero

meansd(x, dig=3)
meansd(x, dig=3, zero_digits=NULL) #or NA
meansd(x, dig=3, only_round=TRUE)
options("crosstable_only_round"=TRUE)
meansd(x, dig=3, zero_digits=2)
options("crosstable_only_round"=NULL)
meanCI(mtcars2$x_date)

#dates
x = as.POSIXct(mtcars$qsec*3600*24 , origin="2010-01-01")
meansd(x)
minmax(x, date_format="%d/%m/%Y")

```

---

test\_correlation\_auto *test for correlation coefficients*

---

**Description**

test for correlation coefficients

**Usage**

```
test_correlation_auto(x, by, method)
```

**Arguments**

x	vector
by	another vector
method	"pearson", "kendall", or "spearman"

**Value**

the correlation test with appropriate method

**Author(s)**

Dan Chaltiel, David Hajage

---

test\_summarize\_auto    *test for mean comparison*

---

**Description**

Compute a oneway.test (with equal or unequal variance) or a kruskal.test as appropriate.

**Usage**

```
test_summarize_auto(x, g)
```

**Arguments**

x	vector
g	another vector

**Value**

a list with two components: p.value and method

**Author(s)**

Dan Chaltiel, David Hajage

---

test\_summarize\_linear\_contrasts  
*Test for linear trend across ordered factor with contrasts*

---

**Description**

Test for linear trend across ordered factor with contrasts

**Usage**

```
test_summarize_linear_contrasts(x, y)
```

**Arguments**

x	vector
y	ordered factor

**Value**

a list with two components: p.value and method



**Author(s)**

Dan Chaltiel

**Examples**

```
library(dplyr)
my_test_args=crosstable_test_args()
my_test_args$test_summarize = test_summarize_linear_contrasts
iris %>%
  mutate(Petal.Width.qt = paste0("Q", ntile(Petal.Width, 5)) %>% ordered()) %>%
  crosstable(Petal.Length ~ Petal.Width.qt, test=TRUE, test_args = my_test_args)
```

---

test\_survival\_logrank *test for survival comparison*

---

**Description**

Compute a logrank test

**Usage**

```
test_survival_logrank(formula)
```

**Arguments**

formula            a formula

**Value**

a list with two components: p.value and method

**Author(s)**

Dan Chaltiel, David Hajage

---

test\_tabular\_auto      *test for contingency table*

---

**Description**

Compute a `chisq.test`, a `chisq.test` with correction of continuity or a fisher test as appropriate

**Usage**

```
test_tabular_auto(x, y)
```

**Arguments**

x                      vector  
y                      another vector

**Value**

a list with two components: `p.value` and `method`

**Author(s)**

Dan Chaltiel, David Hajage

---

transpose\_crosstable      *Transpose a crosstable*

---

**Description**

Pivot a crosstable so the `label` column is swapped with the `by` row. This requires the `variable` column to be the same for every data column, like when all columns are numeric or when all columns are factors with the same levels

**Usage**

```
transpose_crosstable(x)  
  
## S3 method for class 'crosstable'  
t(x)
```

**Arguments**

x                      a crosstable

**Value**

a tibble of class `transposed_crosstable`

**Examples**

```
ct = crosstable(mtcars2, c(mpg, drat, wt, qsec), by=am)
t_ct = t(ct)
as_flextable(t_ct)
```

---

write_and_open	<i>Alternative to default officer print() function. Write the file and try to open it right away.</i>
----------------	---

---

**Description**

As it tests if the file is writable, this function also prevents `officer:::print.rdocx()` to abort the RStudio session.

**Usage**

```
write_and_open(doc, docx.file)
```

**Arguments**

doc	the docx object
docx.file	the name of the target file. If missing or NULL, the doc will open in a temporary file.

**Value**

Nothing, called for its side effects

**Author(s)**

Dan Chaltiel

**Examples**

```
library(officer)
library(crosstable)
mytable = crosstable(mtcars2)
doc = read_docx() %>%
  body_add_crosstable(mytable)

write_and_open(doc)
## Not run:
write_and_open(doc, "example.docx")

## End(Not run)
```

# Index

- \* **as\_gt methods**
  - as\_gt.crosstable, 4
- \* **datasets**
  - iris2, 45
  - mtcars2, 47
  
- apply\_labels, 3
- as\_flextable(), 23, 30
- as\_flextable.crosstable(), 5, 7
- as\_gt (as\_gt.crosstable), 4
- as\_gt.crosstable, 4
- as\_workbook, 5
  
- body\_add\_crosstable, 6
- body\_add\_crosstable(), 6, 8, 17, 19
- body\_add\_crosstable\_footnote, 8
- body\_add\_crosstable\_list
  - (body\_add\_table\_list), 17
- body\_add\_figure\_legend
  - (body\_add\_legend), 10
- body\_add\_figure\_legend(), 12
- body\_add\_flextable(), 17, 19
- body\_add\_flextable\_list
  - (body\_add\_table\_list), 17
- body\_add\_gg2, 8
- body\_add\_gg2(), 10, 30
- body\_add\_glued (body\_add\_normal), 15
- body\_add\_img2, 9
- body\_add\_img2(), 30
- body\_add\_legend, 10
- body\_add\_list, 13
- body\_add\_list(), 31
- body\_add\_list\_item (body\_add\_list), 13
- body\_add\_normal, 15
- body\_add\_normal(), 10–12, 14, 16, 20
- body\_add\_table\_legend
  - (body\_add\_legend), 10
- body\_add\_table\_legend(), 12
- body\_add\_table\_list, 17
- body\_add\_table\_section, 18
  
- body\_add\_title, 19
- body\_replace\_text\_at\_bkms, 21
  
- clean\_names\_with\_labels, 21
- compact (ct\_compact), 34
- confint\_numeric, 22
- copy\_label\_from (set\_label), 52
- cross\_summary, 34
- cross\_summary(), 24
- crosstable, 23
- crosstable(), 6, 26, 29, 30, 32, 53
- crosstable\_effect\_args, 25, 26
- crosstable\_effect\_args(), 36, 38, 40
- crosstable\_options, 27
- crosstable\_options(), 15
- crosstable\_peek\_options, 31
- crosstable\_peek\_options(), 27, 31
- crosstable\_reset\_options, 32
- crosstable\_reset\_options(), 27, 31
- crosstable\_test\_args, 25, 32
- ct\_compact, 34
  
- diff\_mean\_auto (effect\_summary), 38
- diff\_mean\_auto(), 27
- diff\_mean\_boot (effect\_summary), 38
- diff\_mean\_boot(), 27
- diff\_mean\_student (effect\_summary), 38
- diff\_mean\_student(), 27
- diff\_median (effect\_summary), 38
- diff\_median(), 27
- diff\_median\_boot (effect\_summary), 38
- display\_effect, 36
- display\_effect(), 27
- display\_test, 33, 36
- docx\_bookmarks2, 37
  
- effect\_odds\_ratio (effect\_tabular), 39
- effect\_odds\_ratio(), 27
- effect\_relative\_risk (effect\_tabular), 39

- effect\_relative\_risk(), 27
- effect\_risk\_difference
  - (effect\_tabular), 39
- effect\_risk\_difference(), 27
- effect\_summary, 26, 38
- effect\_survival, 26, 39
- effect\_survival\_coxph
  - (effect\_survival), 39
- effect\_survival\_coxph(), 27
- effect\_tabular, 26, 39
- exps::apply\_labels(), 3
- exps::unlab, 51
- exps::var\_lab(), 43
  
- flextable(), 35
- flextable::body\_add\_flextable(), 17
- format.pval(), 50, 51
- format\_fixed, 40
- format\_fixed(), 24, 29, 34, 54
  
- generate\_autofit\_macro, 41
- get\_label, 42, 51
- get\_label(), 45, 53
- get\_percent\_pattern, 43
- ggplot2::ggsave(), 9
- glue::glue(), 24
- gt::gt(), 5
  
- Hmisc::label(), 43
  
- import\_labels, 44, 51
- import\_labels(), 24, 43, 53
- iris, 45
- iris2, 45
- is.compacted\_crosstable
  - (is.crosstable), 46
- is.crosstable, 46
- is.multiby\_crosstable (is.crosstable), 46
- is.transposed\_crosstable
  - (is.crosstable), 46
  
- meanCI (summaryFunctions), 53
- meansd (summaryFunctions), 53
- mediqr (summaryFunctions), 53
- minmax (summaryFunctions), 53
- moystd (summaryFunctions), 53
- mtcars, 47
- mtcars2, 47
  
- N, 48
- na, 48
- narm, 49
- nna (summaryFunctions), 53
  
- officer::body\_add\_gg(), 8
- officer::body\_add\_img(), 9, 10
- officer::body\_add\_par(), 14
- officer::docx\_bookmarks(), 37
- officer::fp\_text(), 11
- officer::fp\_text\_lite(), 11
- officer::read\_docx(), 7, 17
  
- peek, 49
- peek(), 30
- pivot\_crosstable, 50
- plim, 50
  
- remove\_label (remove\_labels), 51
- remove\_label(), 43, 45, 53
- remove\_labels, 51
- rename\_dataframe\_with\_labels
  - (rename\_with\_labels), 52
- rename\_with\_labels, 52
- round(), 54
  
- save\_labels (import\_labels), 44
- save\_labels(), 44, 45
- set\_label, 51, 52
- set\_label(), 24, 43, 45
- stats::confint, 22
- strptime, 24, 41
- summaryFunctions, 53
- survival::Surv(), 24, 39
  
- t.crosstable (transpose\_crosstable), 58
- test\_args (crosstable\_test\_args), 32
- test\_correlation\_auto, 33, 55
- test\_summarize\_auto, 33, 56
- test\_summarize\_linear\_contrasts, 33, 56
- test\_survival\_logrank, 33, 57
- test\_tabular\_auto, 33, 58
- transpose\_crosstable, 58
  
- write\_and\_open, 59