Package 'surveytable'

March 13, 2025

Title Formatted Survey Estimates

Version 0.9.6

Description Short and understandable commands that generate tabulated, formatted, and rounded survey estimates. Mostly a wrapper for the 'survey' package (Lumley (2004) <doi:10.18637/jss.v009.i08> <https://CRAN.R-project.org/package=survey>) that identifies low-precision estimates using the National Center for Health Statistics (NCHS) presentation standards (Parker et al. (2017) <https://www.cdc.gov/nchs/data/series/sr_02/sr02_175.pdf>, Parker et al. (2023) <doi:10.15620/cdc:124368>).

Date/Publication 2025-03-13 08:20:02 UTC

License Apache License (>= 2)

Encoding UTF-8

RoxygenNote 7.3.1

Depends R (>= 2.10)

LazyData true

LazyDataCompression bzip2

Imports assertthat, magrittr, glue, survey, huxtable, lifecycle

Suggests kableExtra, gt, knitr, rmarkdown

VignetteBuilder knitr

URL https://cdcgov.github.io/surveytable/, https://github.com/CDCgov/surveytable

Language en-US

NeedsCompilation no

Author Alex Strashny [aut, cre] (<https://orcid.org/0000-0002-6408-7745>)

Maintainer Alex Strashny <alex.strashny@gmail.com>

Repository CRAN

Contents

---

codebook *Create a codebook for the survey*

Description

Create a codebook for the survey

Usage

```
codebook(all = FALSE, csv = getOption("surveytable.csv"))
```

Arguments

`all`	tabulate all the variables?
`csv`	name of a CSV file

Value

A list of tables.

Examples

```
set_survey(namcs2019sv)
codebook()
```

`namcs2019sv` *Selected variables from the National Ambulatory Medical Care Survey (NAMCS) 2019 Public Use File (PUF)*

Description

Selected variables from a data system of visits to office-based physicians. Note that the unit of observation is visits, not patients - this distinction is important since a single patient can make multiple visits.

Usage

```
namcs2019sv

namcs2019sv_df
```

Format

An object of class `survey.design2` (inherits from `survey.design`) with 8250 rows and 33 columns.

An object of class `data.frame` with 8250 rows and 33 columns.

Details

`namcs2019sv_df` is a data frame.

`namcs2019sv` is a survey object created from `namcs2019sv_df` using `survey::svydesign()`.

Source

- SAS data: https://ftp.cdc.gov/pub/Health_Statistics/NCHS/Dataset_Documentation/NAMCS/sas/namcs2019_sas.zip
- Survey design variables: https://ftp.cdc.gov/pub/Health_Statistics/NCHS/Dataset_Documentation/NAMCS/sas/readme2019-sas.txt
- SAS formats: https://ftp.cdc.gov/pub/Health_Statistics/NCHS/Dataset_Documentation/NAMCS/sas/nam19for.txt
- Documentation: https://ftp.cdc.gov/pub/Health_Statistics/NCHS/Dataset_Documentation/NAMCS/doc2019-508.pdf
- National Summary Tables: https://www.cdc.gov/nchs/data/ahcd/namcs_summary/2019-namcs-web-tables-50pdf

print.surveytable_table *Print surveytable tables*

Description

If a tabulation function is called from the top level, it should print out its table(s) on its own. If that tabulation function is called not from the top level, such as from within a loop or another function, you need to call `print()` explicitly. For example:

```
set_survey(namcs2019sv)
for (vr in c("AGER", "SEX")) {
  print( tab_subset(vr, "MAJOR", "Preventive care") )
}
```

Usage

```
## S3 method for class 'surveytable_table'
print(x, ...)

## S3 method for class 'surveytable_list'
print(x, ...)

as_object(x, ...)
```

Arguments

`x`	an object of class `surveytable_table` or `surveytable_list`.
`...`	passed to helper functions.

Details

The package used to produce the tables can be changed. See `set_opts()` for details. By default, `huxtable` is used.

`as_object()` returns an object (or a list of objects) of whatever package is being used for printing (such as `huxtable`). This is useful for further customizing the tables before finally printing them.

Value

`print.*` returns `x` invisibly.

`as_object()` returns an object (or a list of objects) of whatever package is being used for printing (such as `huxtable`).

Examples

```
set_survey(namcs2019sv)
table1 = tab("AGER")
print(table1)
table_many = tab("MDDO", "SPECCAT", "MSA")
print(table_many)
```

`rccsu2018` *National Study of Long-Term Care Providers (NSLTCP) Residential Care Community (RCC) Services User (SU) 2018 Public Use File (PUF)*

Description

A data system of RCC residents.

Usage

```
rccsu2018
```

Format

An object of class `survey.design2` (inherits from `survey.design`) with 904 rows and 81 columns.

Source

- SAS data: https://ftp.cdc.gov/pub/Health_Statistics/NCHS/Datasets/NPALS/
- Documentation: https://www.cdc.gov/nchs/npals/RCCresident-readme03152021vr.pdf
- Codebook: https://www.cdc.gov/nchs/data/npals/final2018rcc_su_puf_codebook.pdf

---

`set_opts` *Set certain options*

Description

`set_opts()` sets certain options. To view these options, use `show_opts()`. For more advanced control and detailed customization, experienced users can also employ `options()` and `show_options()` (refer to surveytable-options for further information).

Usage

```
set_opts(
  mode = NULL,
  count = NULL,
  lpe = NULL,
  drop_na = NULL,
  max_levels = NULL,
  csv = NULL,
  output = NULL
)

show_opts()
```

Arguments

mode	"general" or "NCHS". See below for details.
count	round counts to the nearest: integer ("int") or one thousand ("1k")
lpe	identify low-precision estimates?
drop_na	drop missing values (NA)? Categorical variables only.
max_levels	a categorical variable can have at most this many levels. Used to avoid printing huge tables.
csv	name of a CSV file or "" to turn off CSV output.
output	package to use for printing. One of "huxtable", "gt", or "kableExtra". For the last two, be sure that this package is installed. "auto" (default) = automatically select huxtable for screen, gt for HTML, or kableExtra for PDF (LaTeX).

Details

If you are not setting a particular option, leave it as NULL.

mode can be either "general" or "NCHS" and has the following meaning:

- "general":
  - Round counts to the nearest integer – same as count = "int".
  - Do not look for low-precision estimates – same as lpe = FALSE.
  - Percentage CI's: use standard Korn-Graubard CI's.
- "nchs":
  - Round counts to the nearest 1,000 – same as count = "1k".
  - Identify low-precision estimates – same as lpe = TRUE.
  - Percentage CI's: adjust Korn-Graubard CI's for the number of degrees of freedom, matching the SUDAAN calculation. NHIS users, be sure to also type options(surveytable.adjust_svyciprop.df_me = "NHIS").

Value

(Nothing.)

See Also

Other options: set_survey(), show_options(), surveytable-options

Examples

```
# Send output to a CSV file:
file_name = tempfile(fileext = ".csv")
suppressMessages( set_opts(csv = file_name) )
set_survey(namcs2019sv)
tab("AGER")
set_opts(csv = "") # Turn off CSV output

show_opts()
```

---

set_survey *Specify the survey to analyze*

Description

You must specify a survey before the other functions, such as tab(), will work. To convert a data.frame to a survey object, see survey::svydesign() or survey::svrepdesign().

Usage

```
set_survey(design, csv = getOption("surveytable.csv"), ...)
```

Arguments

design	either a survey object (created with survey::svydesign() or survey::svrepdesign()); or, for an unweighted survey, a data.frame.
csv	name of a CSV file
...	arguments to set_opts().

Details

Optionally, the survey can have an attribute called label, which is the long name of the survey. Optionally, each variable in the survey can have an attribute called label, which is the variable's long name.

Value

info about the survey

See Also

Other options: set_opts(), show_options(), surveytable-options

Examples

```
set_survey(namcs2019sv)
set_survey(namcs2019sv, mode = "general")
```

`show_options` *Show package options*

Description

See surveytable-options for a discussion of some of the options.

Usage

```
show_options(sw = "surveytable")
```

Arguments

`sw`	starting characters

Value

List of options and their values.

See Also

Other options: `set_opts()`, `set_survey()`, `surveytable-options`

Examples

```
show_options()
```

`surveytable-options` *Package options*

Description

This article is for more advanced users. Typical users, see `set_opts()` and `show_opts()` to set and show certain options.

Details

To view all available options, use show_options(). Below is a description of some noteworthy options.

Changing the number of decimal places or significant digits:

By default, all estimates are rounded in a certain way. The user can change how the rounding is performed.

The following options are the names of functions that control rounding: surveytable.tx_count (for estimates of counts), surveytable.tx_prct (for estimates of percentages), surveytable.tx_rate (for estimates of rates), and surveytable.tx_numeric (for estimates of numeric variables). To turn off all rounding, set each one of these options to ".tx_none".

Each function takes one argument, a data.frame with the following columns: x (point estimates), s (standard errors), ll and ul (CI's). Each function outputs a data.frame with the same column names. For examples of how this works, see the internal functions surveytable:::.tx_count_int (counts, rounded to the nearest integer), surveytable:::.tx_count_1k (counts, rounded to the nearest one thousand), surveytable:::.tx_prct (percentages), surveytable:::.tx_rate (rates), and surveytable:::.tx_numeric (numeric variables).

You can set the above options to your own custom functions. You might also want to adjust the following options, which are the names of columns in the printed tables: surveytable.names_count (by default, this changes when rounding counts to the nearest one thousand) and surveytable.names_prct.

Printing using various table-making packages:

The tabulation functions return objects of class surveytable_table (for a single table) or surveytable_list (for multiple tables, which is just a list of surveytable_table objects). A surveytable_table object is just a data.frame with the following attributes: title, footer, and num, which is the index of columns that should be formatted as a number.

Naturally, these objects can be printed using a variety of packages. surveytable ships with the ability to use huxtable, gt, or kableExtra. See the output argument of set_opts().

You can supply custom code to use another table-making package or to use one of these table-making packages, but in a different way. The two relevant options are surveytable.output_object and surveytable.output_print.

surveytable.output_object is the name of a function with the following arguments: x and ..., where x is a surveytable_table object. This function returns an object from a table-making package, for example, it returns a gt object. Be sure that this package is installed.

surveytable.output_print is the name of a function with the following arguments: x and ..., where x is an object returned by the surveytable.output_object function. The surveytable.output_print function prints this object.

For an example of how this works, see the internal functions surveytable:::.as_object_huxtable and surveytable:::.print_huxtable.

Low-precision estimates:

Optionally, all of the tabulation functions can identify low-precision estimates. Turn on this functionality using any of the following: set_opts(lpe = TRUE), set_opts(mode = "nchs"), set_survey(*, mode = "nchs"), or options(surveytable.find_lpe = TRUE).

By default, low-precision estimates are identified using National Center for Health Statistics (NCHS) algorithms. However, this can be changed, as described below.

Here is a description of the options related to the identification of low-precision estimates.

- `surveytable.find_lpe`: should the tabulation functions look for low-precision estimates? You can change this directly with `options()` or with either `set_opts()` or `set_survey()`.
- `surveytable.lpe_n`, `surveytable.lpe_counts`, `surveytable.lpe_percents`: names of 3 functions.

The argument for `surveytable.lpe_n` is a vector of the number of observations for each level of the variable.

The argument for `surveytable.lpe_counts` is a data frame with count-related estimates. Specifically, the data frame has the following variables:

- `x`: point estimates of counts
- `s`: SE
- `ll`, `ul`: CI
- `samp.size`: effective sample size
- `counts`: actual sample size
- `degf`: degrees of freedom

The argument for `surveytable.lpe_percents` is a data frame with percent-related estimates. Specifically, the data frame has the following variables:

- `Proportion`: point estimates of proportions (between `0` and `1`)
- `SE`: SE
- `LL`, `UL`: CI
- `n numerator`: the number of observations for which the variable is `TRUE`
- `n denominator`: the total number of observations

Each of these functions must return a list with the following elements:

- `id`: the name of the algorithm used, such as `"NCHS presentation standards"`
- `flags`: a vector. For each level of the variable, short codes indicating the presence of low-precision estimates.
- `has.flag`: a vector of short codes that are present in `flags`.
- `descriptions`: a named vector. The names must be the short codes, the values are the longer descriptions.

For example, if a variable has 3 levels, `flags` might be `c("", "A1 A2", "")`. This indicates that for the first and third level, nothing was found, whereas for the second level, two different things were found, indicated by short codes `A1` and `A2`. In this case, `has.flag = c("A1", "A2")`, `descriptions = c(A1 = "A1: something", A2 = "A2: something else")`.

Author(s)

Maintainer: Alex Strashny <alex.strashny@gmail.com> (ORCID)

See Also

Useful links:

- https://cdcgov.github.io/surveytable/
- https://github.com/CDCgov/surveytable

Other options: `set_opts()`, `set_survey()`, `show_options()`

survey_subset *Subset a survey, while preserving variable labels*

Description

Subset a survey, while preserving variable labels

Usage

```
survey_subset(design, subset, label)
```

Arguments

design	a survey object
subset	an expression specifying the sub-population
label	survey label of the newly created survey object

Value

a new survey object

Examples

```
children = survey_subset(namcs2019sv, AGE < 18, "Children < 18")
set_survey(children)
tab("AGER")
```

svyciprop_adjusted *Confidence intervals for proportions, adjusted for degrees of freedom*

Description

A version of survey::svyciprop() that adjusts for the degrees of freedom when method = "beta".

Usage

```
svyciprop_adjusted(
  formula,
  design,
  method = c("logit", "likelihood", "asin", "beta", "mean", "xlogit"),
  level = 0.95,
  df_method,
  ...
)
```

Arguments

formula	see survey::svyciprop().
design	see survey::svyciprop().
method	see survey::svyciprop().
level	see survey::svyciprop().
df_method	how df should be calculated: "default" or "NHIS".
...	see survey::svyciprop().

Details

Written by Makram Talih in 2019.

df_method: for "default", df = degf(design); for "NHIS", df = nrow(design) - 1.

To use this adjustment in surveytable tabulations, call set_survey() or set_opts() with the mode = "NCHS" argument, or type: options(surveytable.adjust_svyciprop = TRUE). NHIS users, be sure to set the surveytable.adjust_svyciprop.df_method option to "NHIS".

Value

The point estimate of the proportion, with the confidence interval as an attribute.

Examples

```
set_survey(namcs2019sv)
set_opts(mode = "NCHS")
tab("AGER")
set_opts(mode = "general")
```

---

tab *Tabulate variables*

Description

Tabulate categorical (factor), logical, or numeric variables.

Usage

```
tab(
  ...,
  test = FALSE,
  alpha = 0.05,
  p_adjust = FALSE,
  drop_na = getOption("surveytable.drop_na"),
  max_levels = getOption("surveytable.max_levels"),
  csv = getOption("surveytable.csv")
)
```

Arguments

...	names of variables (in quotes)
test	perform hypothesis tests?
alpha	significance level for tests
p_adjust	adjust p-values for multiple comparisons?
drop_na	drop missing values (NA)? Categorical variables only.
max_levels	a categorical variable can have at most this many levels. Used to avoid printing huge tables.
csv	name of a CSV file

Details

For categorical and logical variables, presents the estimated counts, their standard errors (SEs) and confidence intervals (CIs), percentages, and their SEs and CIs. Checks the presentation guidelines for counts and percentages and flags estimates if, according to the guidelines, they should be suppressed, footnoted, or reviewed by an analyst.

For numeric variables, presents the percentage of observations with known values, the mean of known values, the standard error of the mean (SEM), and the standard deviation (SD).

CIs are calculated at the 95% confidence level. CIs for count estimates are the log Student's t CIs, with adaptations for complex surveys. CIs for percentage estimates are the Korn and Graubard CIs.

Value

A list of tables or a single table.

See Also

Other tables: tab_cross(), tab_rate(), tab_subset_rate(), total(), total_rate()

Examples

```
set_survey(namcs2019sv)
tab("AGER")
tab("MDDO", "SPECCAT", "MSA")

# Numeric variables
tab("NUMMED")

# Hypothesis testing with categorical variables
tab("AGER", test = TRUE)
```

tab_cross *Tabulate subsets or interactions*

Description

Create subsets of the survey using one variable, and tabulate another variable within each of the subsets. Interact two variables and tabulate.

Usage

```
tab_cross(
  vr,
  vrby,
  max_levels = getOption("surveytable.max_levels"),
  csv = getOption("surveytable.csv")
)

tab_subset(
  vr,
  vrby,
  lvls = c(),
  test = FALSE,
  alpha = 0.05,
  p_adjust = FALSE,
  drop_na = getOption("surveytable.drop_na"),
  max_levels = getOption("surveytable.max_levels"),
  csv = getOption("surveytable.csv")
)
```

Arguments

vr	variable to tabulate
vrby	use this variable to subset the survey
max_levels	a categorical variable can have at most this many levels. Used to avoid printing huge tables.
csv	name of a CSV file
lvls	(optional) only show these levels of vrby
test	perform hypothesis tests?
alpha	significance level for tests
p_adjust	adjust p-values for multiple comparisons?
drop_na	drop missing values (NA)? Categorical variables only.

Details

tab_subset creates subsets using the levels of vrby, and tabulates vr in each subset. Optionally, only use the lvls levels of vrby. vr can be categorical (factor), logical, or numeric.

tab_cross crosses or interacts vr and vrby and tabulates the new variable. Tables created using tab_subset and tab_cross have the same counts but different percentages. With tab_subset, percentages within each subset add up to 100%. With tab_cross, percentages across the entire population add up to 100%. Also see var_cross().

test = TRUE performs a test of association between the two variables. Also performs t-tests for all possible pairs of levels of vr and vrby.

Value

A list of tables or a single table.

See Also

Other tables: tab(), tab_rate(), tab_subset_rate(), total(), total_rate()

Examples

```
set_survey(namcs2019sv)

# For each SEX, tabulate AGER
tab_subset("AGER", "SEX")

# Same counts as tab_subset(), but different percentages.
tab_cross("AGER", "SEX")

# Numeric variables
tab_subset("NUMMED", "AGER")

# Hypothesis testing
tab_subset("NUMMED", "AGER", test = TRUE)
```

---

tab_rate *Calculate rates*

Description

Calculate the rates for categorical (factor) or logical variables.

Usage

```
tab_rate(
  vr,
  pop,
  per = getOption("surveytable.rate_per"),
  drop_na = getOption("surveytable.drop_na"),
```

```
  max_levels = getOption("surveytable.max_levels"),
  csv = getOption("surveytable.csv")
)
```

Arguments

vr	variable to tabulate
pop	either a single number or a data.frame with columns named Level and Population. Level must exactly match the levels of vr. Population is the population for that level of vr.
per	calculate rate per this many items in the population
drop_na	drop missing values (NA)?
max_levels	a categorical variable can have at most this many levels. Used to avoid printing huge tables.
csv	name of a CSV file

Value

A list of tables or a single table.

See Also

Other tables: tab(), tab_cross(), tab_subset_rate(), total(), total_rate()

Examples

```
set_survey(namcs2019sv)
# pop is a data frame
tab_rate("MSA", uspop2019$MSA)

# pop is a single number
tab_rate("MDDO", uspop2019$total)
```

---

tab_subset_rate *Calculate rates for subsets*

Description

Create subsets of the survey using one variable, and tabulate the rates of another variable within each of the subsets.

Usage

```
tab_subset_rate(
  vr,
  vrby,
  pop,
  lvls = c(),
  per = getOption("surveytable.rate_per"),
  drop_na = getOption("surveytable.drop_na"),
  max_levels = getOption("surveytable.max_levels"),
  csv = getOption("surveytable.csv")
)
```

Arguments

vr	variable to tabulate
vrby	use this variable to subset the survey
pop	a data.frame with columns named Level, Subset, and Population. Level must exactly match the levels of vr. Subset must exactly match the levels of vrby. Population is the population for that level of vr and vrby.
lvls	(optional) only show these levels of vrby
per	calculate rate per this many items in the population
drop_na	drop missing values (NA)?
max_levels	a categorical variable can have at most this many levels. Used to avoid printing huge tables.
csv	name of a CSV file

Value

A list of tables or a single table.

See Also

Other tables: tab(), tab_cross(), tab_rate(), total(), total_rate()

Examples

```
set_survey(namcs2019sv)
tab_subset_rate("AGER", "SEX", uspop2019$`AGER x SEX`)
```

total *Total count*

Description

Total count

Usage

```
total(csv = getOption("surveytable.csv"))
```

Arguments

csv	name of a CSV file

Value

A table

See Also

Other tables: tab(), tab_cross(), tab_rate(), tab_subset_rate(), total_rate()

Examples

```
set_survey(namcs2019sv)
total()
```

total_rate *Overall rate*

Description

Overall rate

Usage

```
total_rate(
  pop,
  per = getOption("surveytable.rate_per"),
  csv = getOption("surveytable.csv")
)
```

Arguments

pop	population
per	calculate rate per this many items in the population
csv	name of a CSV file

Value

A table

See Also

Other tables: tab(), tab_cross(), tab_rate(), tab_subset_rate(), total()

Examples

```
set_survey(namcs2019sv)
total_rate(uspop2019$total)
```

---

uspop2019 *US Population in 2019*

Description

Population estimates of the civilian non-institutional population of the United States as of July 1, 2019. Used for calculating rates. For usage examples, see the *_rate functions.

Usage

```
uspop2019
```

Format

An object of class list of length 7.

var_all *Are all the variables true? (Logical AND)*

Description

Create a new variable which is true if all of the variables in a list of variables are true.

Usage

```
var_all(newvr, vrs)
```

Arguments

newvr	name of the new variable to be created
vrs	vector of logical variables

Value

Survey object

See Also

Other variables: var_any(), var_case(), var_collapse(), var_copy(), var_cross(), var_cut(), var_not()

Examples

```
set_survey(namcs2019sv)
var_all("Medicare and Medicaid", c("PAYMCARE", "PAYMCAID"))
tab("Medicare and Medicaid")
```

var_any *Is any variable true? (Logical OR)*

Description

Create a new variable which is true if any of the variables in a list of variables are true.

Usage

```
var_any(newvr, vrs)
```

Arguments

newvr	name of the new variable to be created
vrs	vector of logical variables

Value

Survey object

See Also

Other variables: var_all(), var_case(), var_collapse(), var_copy(), var_cross(), var_cut(), var_not()

Examples

```
set_survey(namcs2019sv)
var_any("Imaging services"
, c("ANYIMAGE", "BONEDENS", "CATSCAN", "ECHOCARD", "OTHULTRA"
, "MAMMO", "MRI", "XRAY", "OTHIMAGE"))
tab("Imaging services")
```

---

var_case *Convert factor to logical*

Description

Convert factor to logical

Usage

```
var_case(newvr, vr, cases, retain_na = TRUE)
```

Arguments

newvr	name of the new logical variable to be created
vr	factor variable
cases	one or more levels of vr that are converted to TRUE. All other levels are converted to FALSE.
retain_na	for the observations where vr is NA, should newvr be NA as well?

Value

Survey object

See Also

Other variables: var_all(), var_any(), var_collapse(), var_copy(), var_cross(), var_cut(), var_not()

Examples

```
set_survey(namcs2019sv)

var_case("Preventive care visits", "MAJOR", "Preventive care")
tab("Preventive care visits")

var_case("Surgery-related visits"
, "MAJOR"
, c("Pre-surgery", "Post-surgery"))
tab("Surgery-related visits")

var_case("Non-primary"
, "SPECCAT.bad"
, c("Surgical care specialty", "Medical care specialty"))
tab("Non-primary")
tab("Non-primary", drop_na = TRUE)
```

---

var_collapse *Collapse factor levels*

Description

Collapse two or more levels of a factor variable into a single level.

Usage

```
var_collapse(vr, newlevel, oldlevels)
```

Arguments

vr	factor variable
newlevel	name of the new level
oldlevels	vector of old levels

Value

Survey object

See Also

Other variables: var_all(), var_any(), var_case(), var_copy(), var_cross(), var_cut(), var_not()

Examples

```
set_survey(namcs2019sv)
tab("PRIMCARE")
var_collapse("PRIMCARE", "Unknown if PCP", c("Blank", "Unknown"))
tab("PRIMCARE")
```

var_copy *Copy a variable*

Description

Create a new variable that is a copy of another variable. You can modify the copy, while the original remains unchanged. See examples.

Usage

```
var_copy(newvr, vr)
```

Arguments

newvr	name of the new variable to be created
vr	variable

Value

Survey object

See Also

Other variables: var_all(), var_any(), var_case(), var_collapse(), var_cross(), var_cut(), var_not()

Examples

```
set_survey(namcs2019sv)
var_copy("Age group", "AGER")
var_collapse("Age group", "65+", c("65-74 years", "75 years and over"))
var_collapse("Age group", "25-64", c("25-44 years", "45-64 years"))
tab("AGER", "Age group")
```

var_cross *Cross or interact two variables*

Description

Create a new variable which is an interaction of two other variables. Also see tab_cross().

Usage

```
var_cross(newvr, vr, vrby)
```

Arguments

newvr	name of the new variable to be created
vr	first variable
vrby	second variable

Value

Survey object

See Also

Other variables: var_all(), var_any(), var_case(), var_collapse(), var_copy(), var_cut(), var_not()

Examples

```
set_survey(namcs2019sv)
var_cross("Age x Sex", "AGER", "SEX")
tab("Age x Sex")
```

---

var_cut *Convert numeric to factor*

Description

Create a new categorical variable based on a numeric variable.

Usage

```
var_cut(newvr, vr, breaks, labels)
```

Arguments

newvr	name of the new factor variable to be created
vr	numeric variable
breaks	see cut()
labels	see cut()

Value

Survey object

See Also

Other variables: var_all(), var_any(), var_case(), var_collapse(), var_copy(), var_cross(), var_not()

Examples

```
set_survey(namcs2019sv)
# In some data systems, variables might contain "special values". For example,
# negative values might indicate unknowns (which should be coded as `NA`).
# Though in this particular data, there are no unknowns.
var_cut("Age group"
  , "AGE"
  , c(-Inf, -0.1, 0, 4, 14, 64, Inf)
  , c(NA, "Under 1", "1-4", "5-14", "15-64", "65 and over"))
tab("Age group")
```

---

var_list *List variables in a survey.*

Description

List variables in a survey.

Usage

```
var_list(sw = "", all = FALSE, csv = getOption("surveytable.csv"))
```

Arguments

sw	starting characters in variable name (case insensitive)
all	print all variables?
csv	name of a CSV file

Value

A table

Examples

```
set_survey(namcs2019sv)
var_list("age")
```

var_not *Logical NOT*

Description

Logical NOT

Usage

```
var_not(newvr, vr)
```

Arguments

newvr	name of the new variable to be created
vr	a logical variable

Value

Survey object

See Also

Other variables: var_all(), var_any(), var_case(), var_collapse(), var_copy(), var_cross(), var_cut()

Examples

```
set_survey(namcs2019sv)
var_not("Private insurance not used", "PAYPRIV")
```

Index