Package ‘wakefield’

May 19, 2018

Title Generate Random Data Sets

Version 0.3.3

Maintainer Tyler Rinker <tyler.rinker@gmail.com>

Description Generates random data sets including: data.frames, lists, and vectors.

Depends R (>= 3.1.2)

Imports chron, ggplot2, dplyr, stringi

Suggests testthat

License GPL-2

LazyData TRUE

URL https://github.com/trinker/wakefield

BugReports https://github.com/trinker/wakefield/issues

Collate 'utils.R' 'r_sample.R' 'age.R' 'r_sample_factor.R' 'animal.R'
'r_sample_binary.R' 'answer.R' 'area.R' 'as_integer.R' 'car.R'
'children.R' 'coin.R' 'color.R' 'date_stamp.R'
'r_sample_logical.R' 'death.R' 'dice.R' 'dna.R' 'dob.R'
'dummy.R' 'education.R' 'employment.R' 'eye.R' 'grade.R'
'grade_level.R' 'group.R' 'hair.R' 'normal.R' 'height.R'
'hour.R' 'id.R' 'income.R' 'internet_browser.R' 'interval.R'
'iq.R' 'language.R' 'level.R' 'r_sample_ordered.R' 'likert.R'
'lorem_ipsum.R' 'marital.R' 'military.R' 'minute.R' 'month.R'
'r_sample_replace.R' 'wakefield-package.R' 'name.R' 'peek.R'
'political.R' 'probs.R' 'r_data.R' 'r_data_frame.R' 'r_dummy.R'
'seriesname.R' 'r_insert.R' 'r_list.R' 'r_na.R'
'r_sample_integer.R' 'r_sample.R' 'race.R' 'relate.R'
'religion.R' 'sat.R' 'second.R' 'sentence.R' 'sex.R'
'sex_inclusive.R' 'smokes.R' 'speed.R' 'state.R' 'string.R'
'table_heat.R' 'time_stamp.R' 'upper.R' 'valid.R' 'variables.R'
'varname.R' 'year.R' 'zip_code.R'

RoxygenNote 6.0.1

NeedsCompilation no
Author  Tyler Rinker [aut, cre],
        Josh O’Brien [ctb],
        Ananda Mahto [ctb],
        Matthew Sigal [ctb],
        Jonathan Carroll [ctb],
        Scott Westenberger [ctb]

Repository  CRAN

Date/Publication  2018-05-19 14:35:10 UTC

R topics documented:

age ................................................................. 4
animal .............................................................. 5
animal_list ....................................................... 6
answer ............................................................. 6
area ................................................................. 7
as_integer ......................................................... 8
car ................................................................. 9
children .......................................................... 10
coin ............................................................... 11
color .............................................................. 12
date_stamp ......................................................... 13
death .............................................................. 14
dice ................................................................. 15
dna ................................................................. 16
dob ................................................................. 17
dummy ............................................................. 18
education ......................................................... 19
employment ...................................................... 20
eye ................................................................. 21
grade ............................................................. 22
grade_level ...................................................... 24
grady_augmented ............................................... 25
group ............................................................. 25
hair ............................................................... 26
height ............................................................ 27
hour ............................................................... 29
id ................................................................. 30
income .......................................................... 31
internet_browser ............................................... 32
interval .......................................................... 33
iq ................................................................. 34
language ........................................................ 35
languages ........................................................ 36
level ............................................................. 37
likert ............................................................ 38
lorem_ipsum .................................................... 39
R topics documented:

marital ................................................................. 40
military ............................................................... 41
minute ................................................................. 42
month ................................................................. 43
name ................................................................. 44
name_neutral .......................................................... 45
normal ................................................................. 45
peek ................................................................. 46
plot.tbl_df ........................................................... 47
political ............................................................ 48
presidential_debates_2012 ....................................... 49
print.available ..................................................... 49
print.variable ...................................................... 50
probs ................................................................. 50
race ................................................................. 51
relate ................................................................. 52
religion ............................................................... 53
r_data ................................................................. 54
r_data_frame ........................................................ 56
r_dummy ............................................................. 58
r_insert ............................................................. 59
r_list ................................................................. 60
r_na ................................................................. 61
r_sample ............................................................ 62
r_sample_binary .................................................... 63
r_sample_factor .................................................... 64
r_sample_integer ................................................... 64
r_sample_logical .................................................. 65
r_sample_ordered .................................................. 66
r_sample_replace .................................................. 67
r_series ............................................................. 68
sat ................................................................. 69
second .............................................................. 71
sentence ............................................................ 72
seriesname .......................................................... 73
sex ................................................................. 73
sex_inclusive ........................................................ 74
smokes .............................................................. 76
speed ............................................................... 77
state ................................................................. 78
state_populations .................................................. 80
string ............................................................... 81
table_heat .......................................................... 82
time_stamp .......................................................... 83
upper ............................................................... 84
valid ................................................................. 85
variables ........................................................... 86
varname ............................................................. 87
Description

Generate a random vector of ages within the provided range. The default age range is set between 18 and 89, to match the age ranges which appear (see e.g., https://gssdataexplorer.norc.org/variables/53/vshow).

Usage

age(n, x = 18:89, prob = NULL, name = "Age")

Arguments

n
The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.

x
A vector of elements to chose from.

prob
A vector of probabilities to chose from.

name
The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random integer vector of ages within the provided range (defaults to 18:89).

See Also

Other variable functions: animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

age(10) # draw 10 ages with default values
hist(age(n=10000))
interval(age, 3, n = 1000)
animal

Generate Random Vector of animals

Description

animal - Generate a random vector of animals.
pet - Generate a random vector of pets.

Usage

animal(n, k = 10, x = wakefield::animal_list, prob = NULL, name = "Animal")

pet(n, x = c(”Dog”, “Cat”, “None”, “Bird”, “Horse”), prob = c(0.365, 0.304, 0.258, 0.031, 0.015), name = "Pet")

Arguments

n  The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
k  The number of the elements of x to sample from (uses sample(x, k)).
x  A vector of elements to chose from.
prob  A vector of probabilities to chose from.
name  The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Details

The household pets and probabilities:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Dog</td>
<td>36.5 %</td>
</tr>
<tr>
<td>Cat</td>
<td>30.4 %</td>
</tr>
<tr>
<td>None</td>
<td>25.8 %</td>
</tr>
<tr>
<td>Bird</td>
<td>3.1 %</td>
</tr>
<tr>
<td>Horse</td>
<td>1.5 %</td>
</tr>
</tbody>
</table>

Value

Returns a random factor vector of animal elements.

See Also

Other variable functions: age, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height,
Examples

animal(10)
pie(table(animal(10000)))

pet(10)
pie(table(pet(10000)))

---

animal_list  Animal List

Description

A dataset containing a character vector animals

Usage

data(animal_list)

Format

A character vector with 591 elements

References

http://a-z-animals.com/animals

---

answer  Generate Random Vector of Answers (Yes/No)

Description

Generate a random vector of answers (yes/no).

Usage

answer(n, x = c("No", "Yes"), prob = NULL, name = "Answer")
Arguments

n  The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
x  A vector of answers to sample from.
prob  A vector of probabilities to chose from.
name  The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random factor vector of answers (yes/no) outcome elements.

See Also

Other variable functions: age, animal, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

answer(10)
100*table(answer(n <- 10000))/n

area  Generate Random Vector of Areas

Description

Generate a random vector of areas ("Suburban", "Urban", "Rural").

Usage

area(n, x = c("Suburban", "Urban", "Rural"), prob = NULL, name = "Area")

Arguments

n  The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
x  A vector of elements to chose from.
prob  A vector of probabilities to chose from.
name  The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.
as_integer

Convert a Factor Data Frame to Integer

Description

Converts a data.frame of factors to integers.

Usage

as_integer(x, cols = NULL, fun = as.integer)

Arguments

x A data.frame of factors.

cols Numeric indices of the columns to include (use - to exclude as well). Default is to assign random NAs to all columns except the first column.

fun An as coercion function to apply to each column. Default is as.integer.

Value

Returns a data.frame equal to the class of x with integer columns rather than factor.

See Also

r_series
Examples

```r
as_integer(r_series(likert_7, 5, 10))
as_integer(r_series(likert_7, 5, 10), cols = c(2, 4))
```

```r
library(dplyr)
r_data_frame(n=100,
  age,
  political,
  sex,
  grade
) %>%
as_integer(2:3)
```

---

**car**  
*Generate Random Vector of Cars*

---

**Description**

Generate a random vector of cars (see ?mtcars).

**Usage**

```r
car(n, x = rownames(datasets::mtcars), prob = NULL, name = "Car")
```

**Arguments**

- `n`  
The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x`  
A vector of elements to chose from.
- `prob`  
A vector of probabilities to chose from.
- `name`  
The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Value**

Returns a random vector of car elements.

**See Also**

Other variable functions: `age`, `animal`, `answer`, `area`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`
Examples

car(10)
table(car(10000))

---

**children**

Generate Random Vector of Number of Children

Description

Generate a random vector of number of children.

Usage

```
children(n, x = 0:10, prob = c(0.25, 0.25, 0.15, 0.15, 0.1, 0.02, 0.02,
    0.02, 0.02, 0.01, 0.01), name = "Children")
```

Arguments

- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x`: A vector of elements to chose from.
- `prob`: A vector of probabilities to chose from.
- `name`: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random vector of number of children elements.

See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

Examples

```
children(10)
pie(table(children(100)))
```
Description

Generate a random vector of coin flips (heads/tails).

Usage

```r
coin(n, x = c("Tails", "Heads"), prob = NULL, name = "Coin")
```

Arguments

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **x**: A vector of coin outcomes to sample from.
- **prob**: A vector of probabilities to chose from.
- **name**: The name to assign to the output vector's `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random factor vector of coin flip outcome elements.

See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

Examples

```r
coin(10)
100*table(coin(n <- 10000))/n
```
color Generate Random Vector of Colors

Description

color - Generate a random vector of colors (sampled from colors()).
color - Generate a random vector of psychological primary colors (sampled from colors()).

Usage

color(n, k = 10, x = grDevices::colors(), prob = NULL, name = "Color")
primary(n, x = c("Red", "Green", "Blue", "Yellow", "Black", "White"),
  prob = NULL, name = "Color")

Arguments

n
The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
k
The number of the elements of x to sample from (uses sample(x, k)).
x
A vector of elements to chose from.
prob
A vector of probabilities to chose from.
name
The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random factor vector of color elements.

See Also

Other variable functions: age, animal, answer, area, car, children, coin, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

color(10)
pie(tab <- table(color(10000)), col = names(tab))

primary(10)
pie(tab <- table(primary(10000)), col = names(tab))
barplot(tab <- table(primary(10000, prob = probs(6))), col = names(tab))
Generate Random Vector of Dates

Description
Generate a random vector of dates.

Usage
```r
date_stamp(n, random = FALSE, x = NULL, start = Sys.Date(), k = 12,
by = "-1 months", prob = NULL, name = "Date")
```

Arguments
- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `random`: logical. If TRUE the dates are randomized, otherwise the dates are sequential.
- `x`: A vector of elements to chose from. This may be NULL if arguments are supplied to `start`, `k`, and `by`. The `x` argument takes precedence over the other three if `is.null` Note that `start`, `k`, and `by` work together to make a vector of dates to sample from. See `seq.Date` for additional information.
- `start`: A date to start the sequence at.
- `k`: The length of the sequence (number of the elements) so build out from `start`.
- `by`: The interval to use in building the sequence.
- `prob`: A vector of probabilities to chose from.
- `name`: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value
Returns a random factor vector of date elements.

See Also
- `seq.Date`
Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`
Examples

date_stamp(10)
pie(table(date_stamp(2000, prob = probs(12))))

## Supply dates to 'x' to sample from
date_stamp(10, x = seq(as.Date("1980-11-16"), length = 30, by = "1 years"))

death

Generate Random Vector of Deaths Outcomes

Description

Generate a random logical vector of deaths (TRUE/FALSE).

Usage

deat(n, prob = NULL, name = "Death")

died(n, prob = NULL, name = "Died")

Arguments

n  The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
prob  A vector of probabilities to chose from.
name  The name to assign to the output vector's varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random logical vector of death outcome elements.

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

dead(10)
died(10)
100*table(death(n <- 10000))/n
100*table(death(n <- 10000, prob = c(.3, .7)))/n
r_data_frame(10, died)
**dice**

Generate Random Vector of Dice Throws

**Description**

Generate a random vector of dice throws.

**Usage**

```r
dice(n, x = 1:6, prob = NULL, name = "Dice")
```

**Arguments**

- `n` The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x` A vector of elements to chose from.
- `prob` A vector of probabilities to chose from.
- `name` The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Value**

Returns a random vector of dice throw elements.

**See Also**

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

**Examples**

```r
dice(10)
barplot(table(dice(10000)))
```
dna

Generate Random Vector of DNA Nucleobases

Description

Generate a random vector of DNA nucleobases ("Guanine", "Adenine", "Thymine", "Cytosine").

Usage

dna(n, x = c("Guanine", "Adenine", "Thymine", "Cytosine"), prob = NULL, name = "DNA")

Arguments

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **x**: A vector of elements to chose from.
- **prob**: A vector of probabilities to chose from.
- **name**: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random vector of DNA nucleobase elements.

See Also

Other variable functions: `age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code`

Examples

dna(10)
barplot(table(dna(10000)))
Generate Random Vector of Birth Dates

Description

Generate a random vector of birth dates.

Usage

```r
dob(n, random = TRUE, x = NULL, start = Sys.Date() - 365 * 15, k = 365 * 2, by = "1 days", prob = NULL, name = "DOB")
```

```r
birth(n, random = TRUE, x = NULL, start = Sys.Date() - 365 * 15, k = 365 * 2, by = "1 days", prob = NULL, name = "Birth")
```

Arguments

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **random**: logical. If `TRUE` the dates are randomized, otherwise the dates are sequential.
- **x**: A vector of elements to chose from. This may be `NULL` if arguments are supplied to `start`, `k`, and `by`. The `x` argument takes precedence over the other three if `!is.null`. Note that `start`, `k`, and `by` work together to make a vector of dates to sample from. See `seq.Date` for additional information.
- **start**: A date to start the sequence at.
- **k**: The length of the sequence (number of the elements) so build out from `start`.
- **by**: The interval to use in building the sequence.
- **prob**: A vector of probabilities to chose from.
- **name**: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random vector of birth date elements.

See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`
**Examples**

dob(10)  
barplot(table(birth(15)))  
barplot(table(birth(30)))

---

**dummy**

*Generate Random Dummy Coded Vector*

**Description**

Generate a random dummy coded (0/1) vector.

**Usage**

dummy(n, prob = NULL, name = "Dummy")

**Arguments**

- **n**: The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
- **prob**: A vector of probabilities to chose from.
- **name**: The name to assign to the output vector's varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

**Value**

Returns a random dummy vector of (0/1) elements.

**See Also**

`sample.int`

Other variable functions: `age, animal, answer, area, car, children, coin, color, date_stap, death, dice, dna, dob, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code`

**Examples**

dummy(100, name = "Var")  
table(dummy(1000))
Generate Random Vector of Educational Attainment Level

**Description**
Generate a random vector of educational attainment level.

**Usage**
```
education(n, x = c("No Schooling Completed", "Nursery School to 8th Grade", "9th Grade to 12th Grade, No Diploma", "Regular High School Diploma", "GED or Alternative Credential", "Some College, Less than 1 Year", "Some College, 1 or More Years, No Degree", "Associate's Degree", "Bachelor's Degree", "Master's Degree", "Professional School Degree", "Doctorate Degree"), prob = c(0.013, 0.05, 0.085, 0.246, 0.039, 0.064, 0.15, 0.075, 0.176, 0.072, 0.019, 0.012), name = "Education")
```

**Arguments**
- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x`: A vector of elements to chose from.
- `prob`: A vector of probabilities to chose from.
- `name`: The name to assign to the output vector's `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Details**
The educational attainments and probabilities used match approximate U.S. educational attainment make-up (http://www.census.gov):

<table>
<thead>
<tr>
<th>Highest Attainment</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Schooling Completed</td>
<td>1.3 %</td>
</tr>
<tr>
<td>Nursery School to 8th Grade</td>
<td>5 %</td>
</tr>
<tr>
<td>9th Grade to 12th Grade, No Diploma</td>
<td>8.5 %</td>
</tr>
<tr>
<td>Regular High School Diploma</td>
<td>24.6 %</td>
</tr>
<tr>
<td>GED or Alternative Credential</td>
<td>3.9 %</td>
</tr>
<tr>
<td>Some College, Less than 1 Year</td>
<td>6.4 %</td>
</tr>
<tr>
<td>Some College, 1 or More Years, No Degree</td>
<td>15 %</td>
</tr>
<tr>
<td>Associate’s Degree</td>
<td>7.5 %</td>
</tr>
<tr>
<td>Bachelor’s Degree</td>
<td>17.6 %</td>
</tr>
<tr>
<td>Master’s Degree</td>
<td>7.2 %</td>
</tr>
<tr>
<td>Professional School Degree</td>
<td>1.9 %</td>
</tr>
<tr>
<td>Doctorate Degree</td>
<td>1.2 %</td>
</tr>
</tbody>
</table>
Value

Returns a random vector of educational attainment level elements.

References

http://www.census.gov

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

education(10)
pie(table(education(10000)))

---

**employment**

*Generate Random Vector of Employment Statuses*

Description

Generate a random vector of employment statuses.

Usage

```r
employment(n, x = c("Full Time", "Part Time", "Unemployed", "Retired", "Student"), prob = c(0.6, 0.1, 0.1, 0.1, 0.1), name = "Employment")
```

Arguments

- `n` - The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x` - A vector of elements to chose from.
- `prob` - A vector of probabilities to chose from.
- `name` - The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.
Details

The following arbitrary probabilities are used:

<table>
<thead>
<tr>
<th>Employment Status</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Time</td>
<td>60%</td>
</tr>
<tr>
<td>Part Time</td>
<td>10%</td>
</tr>
<tr>
<td>Unemployed</td>
<td>10%</td>
</tr>
<tr>
<td>Retired</td>
<td>10%</td>
</tr>
<tr>
<td>Student</td>
<td>10%</td>
</tr>
</tbody>
</table>

Value

Returns a random vector of employment status elements.

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

employment(10)
pie(table(employment(10000)))
barplot(table(employment(10000)))

Description

Generate a random vector of eye colors.

Usage

eye(n, x = c("Brown", "Blue", "Green", "Hazel", "Gray"), prob = c(0.44, 0.3, 0.13, 0.09, 0.04), name = "Eye")

Arguments

n The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.

x A vector of elements to chose from.

prob A vector of probabilities to chose from.
name

The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Details

The eye colors and probabilities:

<table>
<thead>
<tr>
<th>Color</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brown</td>
<td>44 %</td>
</tr>
<tr>
<td>Blue</td>
<td>30 %</td>
</tr>
<tr>
<td>Green</td>
<td>13 %</td>
</tr>
<tr>
<td>Hazel</td>
<td>9 %</td>
</tr>
<tr>
<td>Gray</td>
<td>4 %</td>
</tr>
</tbody>
</table>

Value

Returns a random vector of eye color elements.

See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

Examples

eye(10)

barplot(v <- table(eye(10000)), col = replace(names(v), 4, "yellowgreen"))

---

grade

Generate Random Vector of Grades

Description

`grade` - Generate a random normal vector of percent grades.

`grade_letter` - Generate a random normal vector of letter grades.

`gpa` - Generate a random normal vector of grade point averages (GPA; 0.0 - 4.0 scale).

Usage

```
grade(n, mean = 88, sd = 4, name = "Grade", digits = 1)

grade_letter(n, mean = 88, sd = 4, name = "Grade_Letter")

gpa(n, mean = 88, sd = 4, name = "GPA")
```
Arguments

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **mean**: The mean value for the normal distribution to be drawn from.
- **sd**: The standard deviation of the normal distribution to draw from.
- **name**: The name to assign to the output vector’s varname attribute. This is used to automatically assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.
- **digits**: Integer indicating the number of decimal places to be used. Negative values are allowed (see `round`).

Details

The conversion between percent range, letter grade, and GPA is:

<table>
<thead>
<tr>
<th>Percent</th>
<th>Letter</th>
<th>GPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>97-100</td>
<td>A+</td>
<td>4.00</td>
</tr>
<tr>
<td>93-96</td>
<td>A</td>
<td>4.00</td>
</tr>
<tr>
<td>90-92</td>
<td>A-</td>
<td>3.67</td>
</tr>
<tr>
<td>87-89</td>
<td>B+</td>
<td>3.33</td>
</tr>
<tr>
<td>83-86</td>
<td>B</td>
<td>3.00</td>
</tr>
<tr>
<td>80-82</td>
<td>B-</td>
<td>2.67</td>
</tr>
<tr>
<td>77-79</td>
<td>C+</td>
<td>2.33</td>
</tr>
<tr>
<td>73-76</td>
<td>C</td>
<td>2.00</td>
</tr>
<tr>
<td>70-72</td>
<td>C-</td>
<td>1.67</td>
</tr>
<tr>
<td>67-69</td>
<td>D+</td>
<td>1.33</td>
</tr>
<tr>
<td>63-66</td>
<td>D</td>
<td>1.00</td>
</tr>
<tr>
<td>60-62</td>
<td>D-</td>
<td>0.67</td>
</tr>
<tr>
<td>&lt; 60</td>
<td>F</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Value

Returns a random normal vector of grade elements.

See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

Examples

```r
grade(10)
hist(grade(10000))
interval(grade, 5, n = 1000)
```
grade_level(10)
barplot(table(grade_level(10000)))

gpa(10)
hist(gpa(10000))

---

**grade_level**

*Generate Random Vector of Grade Levels*

**Description**

Generate a random vector of grade levels.

**Usage**

```r
grade_level(n, x = c("K", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12"), prob = NULL, name = "Grade_Level")
```

**Arguments**

- `n` The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x` A vector of elements to chose from.
- `prob` A vector of probabilities to chose from.
- `name` The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Value**

Returns a random vector of grade level elements.

**See Also**

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

**Examples**

```r
grade_level(10)
barplot(table(grade_level(10000)))
```
**grady_augmented**  

**Augmented List of Grady Ward’s English Words and Mark Kantrowitz’s Names List**

**Description**

A dataset containing a vector of Grady Ward’s English words augmented with `qdapDictionaries`’s DICTIONARY, Mark Kantrowitz’s names list, other proper nouns, and contractions.

**Usage**

```r
data(grady_augmented)
```

**Format**

A character vector with 122806 elements

**Details**

A dataset containing a vector of Grady Ward’s English words augmented with proper nouns (U.S. States, Countries, Mark Kantrowitz’s Names List, and months) and contractions. That dataset is augmented to increase the data set size.

**References**

Moby Thesaurus List by Grady Ward [http://www.gutenberg.org](http://www.gutenberg.org)

List of names from Mark Kantrowitz [http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/nlp/corpora/names/]. A copy of the README is available here per the author’s request.

---

**group**  

**Generate Random Vector of Control/Treatment Groups**

**Description**

Generate a random vector of binary groups (e.g., control/treatment).

**Usage**

```r
group(n, x = c("Control", "Treatment"), prob = NULL, name = "Group")
```
Arguments

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **x**: A vector of groups to sample from.
- **prob**: A vector of probabilities to chose from.
- **name**: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random factor vector of group (control/treatment) elements.

Note

If you want > 2 groups see ‘r_sample_factor’.

See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

Examples

```r
group(10)
100*table(group(n <- 10000))/n
100*table(group(n <- 10000, prob = c(.3,.7)))/n
```

---

**hair**

*Generate Random Vector of Hair Colors*

Description

Generate a random vector of hair colors.

Usage

```r
hair(n, x = c("Brown", "Black", "Blonde", "Red"), prob = c(.35, .28, .26, .11), name = "hair")
```
Arguments

- **n**: The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
- **x**: A vector of elements to chose from.
- **prob**: A vector of probabilities to chose from.
- **name**: The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Details

The hair colors and probabilities:

<table>
<thead>
<tr>
<th>Color</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brown</td>
<td>35 %</td>
</tr>
<tr>
<td>Black</td>
<td>28 %</td>
</tr>
<tr>
<td>Blonde</td>
<td>26 %</td>
</tr>
<tr>
<td>Red</td>
<td>11 %</td>
</tr>
</tbody>
</table>

Value

Returns a random vector of hair color elements.

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

```r
hair(10)
v <- table(hair(10000))
lbs <- paste0(names(v), "\n", round(100*v/sum(v), 1), "\%")
pie(v, col = replace(names(v), 3, "yellow"), labels = lbs)
```

---

### height

**Generate Random Vector of Heights**

**Description**

- **height** and **height_in** - Generate a random normal vector of heights in inches.
- **height_cm** - Generate a random normal vector of heights in centimeters.
Usage

```r
height(n, mean = 69, sd = 3.75, min = 1, max = NULL, digits = 0,
    name = "Height")

height_in(n, mean = 69, sd = 3.75, min = 1, max = NULL, digits = 1,
    name = "Height(in")

height_cm(n, mean = 175.26, sd = 9.525, min = 1, max = NULL,
    digits = 1, name = "Height(cm")
```

Arguments

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **mean**: The mean value for the normal distribution to be drawn from.
- **sd**: The standard deviation of the normal distribution to draw from.
- **min**: A numeric lower boundary cutoff. Results less than this value will be replaced with `min`.
- **max**: A numeric upper boundary cutoff. Results greater than this value will be replaced with `max`.
- **digits**: Integer indicating the number of decimal places to be used. Negative values are allowed (see `round`).
- **name**: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random normal vector of height elements.

Note

height rounds to nearest whole number. height_in & height_in round to the nearest tenths.

See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

Examples

```r
height(10)
hist(height(10000))
interval(height, 5, n = 1000)
```
hour  

Generate a Random Sequence of H:M:S Times

Description

Generate a random vector of H:M:S times.

Usage

```r
hour(n, x = seq(0, 23.5, by = 0.5), prob = NULL, random = FALSE, name = "Hour")
```

Arguments

- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x`: A vector of elements to chose from.
- `prob`: A vector of probabilities to chose from.
- `random`: logical. If TRUE the times are randomized, otherwise the times are sequential.
- `name`: The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random vector of H:M:S time elements.

See Also

`times`

Examples

```r
hour(20)
hour(20, random=TRUE)
```
id - Generate a sequential character vector of zero-padded identification numbers (IDs).
id_factor - Generate a sequential factor vector of zero-padded identification numbers (IDs).

Usage

id(n, random = FALSE, name = "ID")

id_factor(n, random = FALSE, name = "ID")

Arguments

n The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
random logical. If TRUE the IDs are randomized, otherwise the IDs are sequential.
name The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a (optionally random) vector of character/factor observations ID numbers.

Warning

id uses sprintf to generate the padded ID. Per sprintf’s documentation: “The format string is passed down the OS’s sprintf function...The behaviour on inputs not documented here is ‘undefined’, which means it is allowed to differ by platform.” See sprintf for details.

Note

id is faster than id_factor, as the later coerces the vector to a factor.

See Also

sprintf

Examples

id(1000)
r_data_frame(n=21, id)
Generate Random Gamma Vector of Incomes

Description

Generate a random gamma vector of incomes.

Usage

```r
income(n, digits = 2, name = "Income")
```

Arguments

- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `digits`: Integer indicating the number of decimal places to be used. Negative values are allowed (see `round`).
- `name`: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Details

Incomes are generated using: `rgamma(n, 2) * 2000`.

Value

Returns a random gamma vector of income elements.

See Also

gamma

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

Examples

```r
income(10)
or hist(income(10000))
or pie(table(cut(income(10000), 10)))
```
internet_browser

Generate Random Vector of Internet Browsers

Description

Generate a random vector of Internet browser.

Usage

internet_browser(n, x = c("Chrome", "IE", "Firefox", "Safari", "Opera", "Android"), prob = c(0.5027, 0.175, 0.1689, 0.0994, 0.017, 0.0132), name = "Browser")

Arguments

n The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
x A vector of elements to chose from.
prob A vector of probabilities to chose from.
name The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Details

The browser use and probabilities (from http://gs.statcounter.com):

<table>
<thead>
<tr>
<th>Browser</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chrome</td>
<td>50.27 %</td>
</tr>
<tr>
<td>IE</td>
<td>17.50 %</td>
</tr>
<tr>
<td>Firefox</td>
<td>16.89 %</td>
</tr>
<tr>
<td>Safari</td>
<td>9.94 %</td>
</tr>
<tr>
<td>Opera</td>
<td>1.70 %</td>
</tr>
<tr>
<td>Android</td>
<td>1.32 %</td>
</tr>
</tbody>
</table>

Value

Returns a random factor vector of Internet browser elements.

References

http://www.pewforum.org/2012/12/18/table-religious-composition-by-country-in-numbers
See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

```r
internet_browser(20)
barplot(table(internet_browser(10000)))
pie(table(internet_browser(10000)))
```

Description

A wrapper for `cut` that cuts the vector and then adds the varname produced by the original function.

Usage

```r
interval(fun, breaks, ..., labels = NULL, include.lowest = FALSE,
      right = TRUE, dig.lab = 3, ordered_result = FALSE, n)
```

Arguments

- **fun**: A vector producing function.
- **breaks**: Either a numeric vector of two or more unique cut points or a single number (greater than or equal to 2) giving the number of intervals into which the vector produced from `fun` is to be cut.
- **labels**: Labels for the levels of the resulting category. By default, labels are constructed using "(a,b]" interval notation. If `labels = FALSE`, simple integer codes are returned instead of a factor.
- **include.lowest**: logical. If TRUE an 'x[i]' equal to the lowest (or highest, for `right = FALSE) 'breaks' value should be included.
- **right**: logical. If TRUE the intervals will be closed on the right (and open on the left).
- **dig.lab**: An integer which is used when labels are not given. It determines the number of digits used in formatting the break numbers.
- **ordered_result**: logical. If TRUE the result be an ordered factor.
- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **...**: Other arguments passed to `fun`.
**Value**

Returns a `cut` factor vector.

**See Also**

`cut`

**Examples**

```r
interval(normal, 4, n=100)
attributes(interval(normal, 4, n=100))
interval(age, 3, n = 1000)
```

---

**iq**

*Generate Random Vector of Intelligence Quotients (IQs)*

**Description**

Generate a random normal vector of intelligence quotients (IQs).

**Usage**

```r
iq(n, mean = 100, sd = 10, min = 0, max = NULL, digits = 0, name = "IQ")
```

**Arguments**

- `n` The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `mean` The mean value for the normal distribution to be drawn from.
- `sd` The standard deviation of the normal distribution to draw from.
- `min` A numeric lower boundary cutoff. Results less than this value will be replaced with `min`.
- `max` A numeric upper boundary cutoff. Results greater than this value will be replaced with `max`.
- `digits` Integer indicating the number of decimal places to be used. Negative values are allowed (see `round`).
- `name` The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Value**

Returns a random normal vector of IQ elements.
language

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

```r
iq(10)
hist(iq(10000))
interval(iq, 5, n = 1000)
```

**language**

*Generate Random Vector of Languages*

Description

Generate a random vector of languages from the presidential.debates.2012.

Usage

```r
language(n, x = wakefield::languages["Language"],
    prob = wakefield::languages["Proportion"], name = "Language")
```

Arguments

- `n` The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
- `x` A vector of elements to chose from.
- `prob` A vector of probabilities to chose from.
- `name` The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random character vector of language elements.

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code
Examples

```r
language(10)
pie(table(language(10000)))

lang <- wakefield::languages[sample(1:99, 6), ]
lang["prop"] <- lang["N"] / sum(lang["N"])
labs <- round(100 * lang["prop"], 1)
pie(lang["prop"], paste0(lang["Language"], 
  \n, labs, ")
```

---

**languages**

<table>
<thead>
<tr>
<th>Languages of the World</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

**Description**


**Usage**

`data(languages)`

**Format**

A data frame with 99 rows and 4 variables

**Details**

- Language. The language spoken.
- N. The number of speakers world-wide.
- Proportion. The proportion of speakers.
- Percent. The percentage of speakers.

**References**

level

Generate Random Vector of Levels

Description

level - Generate a random vector of integer levels (1-4).

math - Generate a random vector of integer mathematics levels (1-4) similar to New York State grades 3-8 assessment results.

ela - Generate a random vector of integer English language arts (ELA) levels (1-4) similar to New York State grades 3-8 assessment results.

Usage

level(n, x = 1:4, prob = NULL, name = "Level")

math(n, x = 1:4, prob = c(0.29829, 0.33332, 0.22797, 0.14042),
    name = "Math")

ela(n, x = 1:4, prob = c(0.3161, 0.37257, 0.2233, 0.08803), name = "ELA")

Arguments

n The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.

x A vector of elements to chose from.

prob A vector of probabilities to chose from.

name The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Details

Distribution of levels (used in prob) were taken from New York State’s 2014 assessment report:

<table>
<thead>
<tr>
<th>Level</th>
<th>ELA</th>
<th>Math</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>31.6%</td>
<td>29.8%</td>
</tr>
<tr>
<td>2</td>
<td>37.3%</td>
<td>33.3%</td>
</tr>
<tr>
<td>3</td>
<td>22.3%</td>
<td>22.8%</td>
</tr>
<tr>
<td>4</td>
<td>8.8%</td>
<td>14.0%</td>
</tr>
</tbody>
</table>

Value

Returns a random vector of integer levels (1-4) elements.
See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

```r
level(10)
barplot(table(level(10000, prob = probs(4))))

math(10)
barplot(table(math(10000)))

ela(10)
barplot(table(ela(10000)))
```

---

**likert**

*Generate Random Vector of Likert-Type Responses*

**Description**

Generate a random vector of Likert-type responses.

**Usage**

```r
likert(n, x = c("Strongly Agree", "Agree", "Neutral", "Disagree", "Strongly Disagree"), prob = NULL, name = "likert")
likert_5(n, x = c("Strongly Agree", "Agree", "Neutral", "Disagree", "Strongly Disagree"), prob = NULL, name = "likert")
likert_7(n, x = c("Strongly Agree", "Agree", "Somewhat Agree", "Neutral", "Somewhat Disagree", "Disagree", "Strongly Disagree"), prob = NULL, name = "likert")
```

**Arguments**

- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x`: A vector of elements to chose from.
- `prob`: A vector of probabilities to chose from.
- `name`: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.
Value

Returns a random vector of Likert-type response elements.

Note

likert & likert_5 are identical outputs, sampling from a 5-point response scale. likert_7 samples from a 7-point response scale.

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

dice(10)
barplot(table(dice(10000)))

---

Value

Returns a random character vector of string elements.
Note

lorem_ipsum and paragraph produce identical strings but will produce different vector/column names when used inside of r_data_frame or r_list.

See Also

stri_rand_lipsum

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

```r
lorem_ipsum(10)
paragraph(10)
lorem_ipsum(10, start_lipsum = FALSE)
```

---

marital

*Generate Random Vector of Marital Statuses*

Description

Generate a random vector of marital statuses.

Usage

```r
marital(n, x = c("Married", "Divorced", "Widowed", "Separated", "Never Married"), prob = NULL, name = "Marital")
```

Arguments

- **n**
  The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
- **x**
  A vector of elements to chose from.
- **prob**
  A vector of probabilities to chose from.
- **name**
  The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random vector of marital status elements.
See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

Examples

```r
marital(10)
barplot(table(marital(10000)))
```

---

**military**

Generate Random Vector of Military Branches

**Description**

Generate a random vector of military branches.

**Usage**

```r
military(n, x = c("Army", "Air Force", "Navy", "Marine Corps", "Coast Guard"),
prob = c(0.3785, 0.2334, 0.2218, 0.1366, 0.0296), name = "Military")
```

**Arguments**

- `n` The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x` A vector of elements to chose from.
- `prob` A vector of probabilities to chose from.
- `name` The name to assign to the output vector's `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Details**

The military branches and probabilities used match approximate U.S. military make-up:

<table>
<thead>
<tr>
<th>Branch</th>
<th>N</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Army</td>
<td>541,291</td>
<td>37.9%</td>
</tr>
<tr>
<td>Air Force</td>
<td>333,772</td>
<td>23.3%</td>
</tr>
<tr>
<td>Navy</td>
<td>317,237</td>
<td>22.2%</td>
</tr>
<tr>
<td>Marine Corps</td>
<td>195,338</td>
<td>13.7%</td>
</tr>
<tr>
<td>Coast Guard</td>
<td>42,357</td>
<td>3.0%</td>
</tr>
</tbody>
</table>
minute

Value

Returns a random factor vector of military branch elements.

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

minute(10)
barplot(table(military(10000)))
pie(table(military(10000)))

minute(n, x = seq(0, 59, by = 1)/60, prob = NULL, random = FALSE, name = "Minute")

Description

Generate a random vector of minutes in H:M:S format.

Usage

minute(n, x = seq(0, 59, by = 1)/60, prob = NULL, random = FALSE, name = "Minute")

Arguments

n
The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.

x
A vector of elements to chose from.

prob
A vector of probabilities to chose from.

random
logical. If TRUE the times are randomized, otherwise the times are sequential.

name
The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random vector of minute time elements in H:M:S format.

See Also
times
Examples

```r
month(20)
month(20, random=TRUE)
pie(table(minute(2000, x = seq(0, 59, by = 10)/60, prob = probs(6))))
```

---

**Generate Random Vector of Months**

**Description**

Generate a random factor vector of months.

**Usage**

```r
month(n, x = month.name, prob = NULL, name = "Month")
```

**Arguments**

- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x`: A vector of elements to chose from.
- `prob`: A vector of probabilities to chose from.
- `name`: The name to assign to the output vector's `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Value**

Returns a random character vector of month elements.

**See Also**

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

**Examples**

```r
month(10)
pie(table(month(10000, prob = probs(12))))
```
Generate Random Vector of Names

Description

Generate a random vector of first names. This dataset includes all unique entries from the babynames package.

Usage

```r
ame(n, x = wakefield::name_neutral, prob = NULL, replace = FALSE, name = "Name")
```

Arguments

- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x`: A vector of elements to chose from.
- `prob`: A vector of probabilities to chose from.
- `replace`: logical. If TRUE sampling is done with replacement. Default is without replacement.
- `name`: The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random vector of name elements.

See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

Examples

```r
name(10)
name(100)
name(1000, replace = TRUE)
```
Gender Neutral Names

Description

A dataset containing a character vector gender neutral names according to the U.S. Census.

Usage

data(name_neutral)

Format

A character vector with 662 elements

References

http://www.census.gov

Generate Random Normal Vector

Description

normal - A wrapper for \texttt{rnorm} that generate a random normal vector.

normal_round - A wrapper for \texttt{rnorm} that generate a rounded random normal vector.

Usage

\texttt{normal}(n, mean = 0, sd = 1, min = \texttt{NULL}, max = \texttt{NULL}, name = "Normal")

\texttt{normal_round}(n, mean = 0, sd = 1, min = \texttt{NULL}, max = \texttt{NULL}, digits = 2, name = "Normal")

Arguments

\begin{itemize}
  \item \texttt{n} \hspace{1cm} The number elements to generate. This can be globally set within the environment of \texttt{r_data_frame} or \texttt{r_list}.
  \item \texttt{mean} \hspace{1cm} The mean value for the normal distribution to be drawn from.
  \item \texttt{sd} \hspace{1cm} The standard deviation of the normal distribution to draw from.
  \item \texttt{min} \hspace{1cm} A numeric lower boundary cutoff. Results less than this value will be replaced with \texttt{min}.
  \item \texttt{max} \hspace{1cm} A numeric upper boundary cutoff. Results greater than this value will be replaced with \texttt{max}.
\end{itemize}
name  The name to assign to the output vector’s varname attribute. This is used to automatically assign names to the column/vector name when used inside of r_data_frame or r_list.

digits  Integer indicating the number of decimal places to be used. Negative values are allowed (see round).

Value

Returns a random vector of elements.

See Also

rnorm
round

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

normal(100, name = "Var")
hist(normal(10000, 100, 10))
interval(normal, 9, n = 1000)

Description

Convenience function to view all the columns of the head of a truncated data.frame. peek invisibly returns x. This makes its use ideal in a dplyr/magrittr pipeline.

Usage

peek(x, n = 10, width = 10, ...)

Arguments

x  A data.frame object.
n  Number of rows to display.
width  The width of the columns to be displayed.
...  For internal use.
plot.tbl_df

Details

By default dplyr does not print all columns of a data frame (tbl_df). This makes inspection of
data difficult at times, particularly with text string data. peek allows the user to see a truncated head
for inspection purposes.

Value

Prints a truncated head but invisibly returns x.

See Also

head

Examples

(dat1 <- r_data_frame(100, id, sentence, paragraph))
peek(dat1)
peek(dat1, n = 20)
peek(dat1, width = 40)

library(dplyr)

## Use in a dplyr/magrittr pipeline to view the data (silly example)
par(mfrow = c(2, 2))
r_data_frame(1000, id, sex, pet, employment, eye, sentence, paragraph) %>%
  peek %>%
  (function(x, ind = 2:5){ invisible(lapply(ind, function(i) pie(table(x[[i]]))))})

## A wider data set example
dat2 <- r_data_theme()

dat2
peek(dat2)

---

**plot.tbl_df**

Plots a *tbl_df* Object

Description

Plots a *tbl_df* object.

Usage

## S3 method for class 'tbl_df'
plot(x, ...)

### political

**Generate Random Vector of Political Parties**

**Description**

Generate a random vector of political parties.

**Usage**

```r
political(n, x = c("Democrat", "Republican", "Constitution", "Libertarian", "Green"), prob = c(0.577269133302994, 0.410800432748879, 0.004918084954793489, 0.00372590303330866, 0.0032936813677832), name = "Political")
```

**Arguments**

- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x`: A vector of elements to chose from.
- `prob`: A vector of probabilities to chose from.
- `name`: The name to assign to the output vector's `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Details**

The political parties and probabilities used match approximate U.S. political make-up of registered voters (2014). The default make up is:

<table>
<thead>
<tr>
<th>Party</th>
<th>N</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Democrat</td>
<td>43,140,758</td>
<td>57.73%</td>
</tr>
<tr>
<td>Republican</td>
<td>30,700,138</td>
<td>41.08%</td>
</tr>
<tr>
<td>Constitution</td>
<td>367,000</td>
<td>.49%</td>
</tr>
<tr>
<td>Libertarian</td>
<td>278,446</td>
<td>.37%</td>
</tr>
<tr>
<td>Green</td>
<td>246,145</td>
<td>.33%</td>
</tr>
</tbody>
</table>

**Value**

Returns a random factor vector of political party elements.
See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

```r
political(10)
barplot(table(political(10000)))
```

---

**presidential_debates_2012**

*2012 U.S. Presidential Debate Dialogue*

Description

A dataset containing 2911 ordered sentences used by speakers during the three 2012 presidential debates.

Usage

```r
data(presidential_debates_2012)
```

Format

A character vector with 2911 elements

---

**print.available**

*Prints an available Object.*

Description

Prints an available object.

Usage

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

Arguments

- `x`: The available object
- `...`: ignored
**print.variable**  
*Prints a variable Object*

**Description**

Prints a variable object

**Usage**

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

**Arguments**

- `x` The variable object.
- `...` Ignored.

---

**probs**  
*Generate a Random Vector of Probabilities.*

**Description**

Generate a random vector of probabilities that sum to 1.

**Usage**

```r
probs(j, upper = 1e+06)
```

**Arguments**

- `j` An integer of number of probability elements (typically performs best at j < 4000).
- `upper` `probs` works by sampling from 1:upper j times and then dividing each sample by the sum of all samples.

**Value**

Returns a vector of probabilities summing to 1.

**Examples**

```r
probs(10)
sum(probs(100))
pie(table(month(10000, prob = probs(12))))
```
race

Generate Random Vector of Races

Description
Generate a random vector of races.

Usage
race(n, x = c("White", "Hispanic", "Black", "Asian", "Bi-Racial", "Native", "Other", "Hawaiian"), prob = c(0.637, 0.163, 0.122, 0.047, 0.019, 0.007, 0.002, 0.0015), name = "Race")

Arguments
n The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
x A vector of elements to chose from.
prob A vector of probabilities to chose from.
name The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Details
The races and probabilities used match approximate U.S. racial make-up. The default make up is:

<table>
<thead>
<tr>
<th>Race</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>White</td>
<td>63.70 %</td>
</tr>
<tr>
<td>Hispanic</td>
<td>16.30 %</td>
</tr>
<tr>
<td>Black</td>
<td>12.20 %</td>
</tr>
<tr>
<td>Asian</td>
<td>4.70 %</td>
</tr>
<tr>
<td>Bi-Racial</td>
<td>1.90 %</td>
</tr>
<tr>
<td>Native</td>
<td>.70 %</td>
</tr>
<tr>
<td>Other</td>
<td>.20 %</td>
</tr>
<tr>
<td>Hawaiian</td>
<td>.15 %</td>
</tr>
</tbody>
</table>

Value
Returns a random factor vector of elements.

See Also
Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair,
height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

```r
code_example
```

relate

Create Related Numeric Columns

Description

Generate columns that are related.

Usage

```r
relate(x, j, name = NULL, operation = "+", mean = 5, sd = 1, rep.sep = "_", digits = max(nchar(sub("^[.].","","", x))))
```

Arguments

- `x`: A starting column.
- `j`: The number of columns to produce.
- `name`: An optional prefix name to give to the columns. If `NULL` attempts to take from the `varname` attribute of `x`. If not found, "Variable" is used.
- `operation`: A operation character vector of length 1; either c("+", ",", ",", "/"). This is the relationship between columns.
- `mean`: Mean is the average value to add, subtract, multiple, or divide by.
- `sd`: The amount of variability to allow in `mean`. Setting to 0 will constrain the operation between `x_(n - 1)` column and `x_n` to be exactly the mean value (see Examples for a demonstration).
- `rep.sep`: A separator to use for repeated variable names. For example if the `age` is used three times (`r_data_frame(age, age, age)`), the name "Age" will be assigned to all three columns. The results in column names c("Age_1", "Age_2", "Age_3").
- `digits`: The number of digits to round to. Defaults to the max number of significant digits in `x`.

Value

Returns a `tbl_df`.

See Also

- `r_series`
**Examples**

```
relate(1:10, 10)
(x <- r_data_frame(10, id, relate(1:10, 10, "Time", mean = 2)))
library(ggplot2)

dat <- with(x, data.frame(ID = rep(ID, ncol(x[, -1])), stack(x[, -1])))
bd <- subset(bd, bd[, -1])
bt <- with(bd, factor(sub("Time", ",", bd[, "ind"])), levels = 1:10)

ggplot(dat, aes(x = Time, y = values, color = ID, group = ID)) +
geom_line(size = 1)
relate(1:10, 10, name = "X", operation = "-")
relate(1:10, 10, "X", mean = 1, sd = 0)
relate(1:10, 10, "Var", "+")
relate(1:10, 10, "Var", "/")
relate(gpa(30), 5, mean = .1)
relate(likert(10), 5, mean = .1, sd = .2)
relate(date_stamp(10), 6)
relate(time_stamp(10), 6)
relate(rep(100, 10), 6, "Reaction", "-")
```

**Description**

`religion` - Generate Random Vector of Religions

Generate a random vector of religion.

**Usage**

```
religion(n, x = c("Christian", "Muslim", "None", "Hindu", "Buddhist", "Folk", "Other", "Jewish"), prob = c(0.31477, 0.23163, 0.16323, 0.14985, 0.07083, 0.05882, 0.00859, 0.00227), name = "Religion")
```

**Arguments**

- `n` The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x` A vector of elements to chose from.
- `prob` A vector of probabilities to chose from.
- `name` The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Details**

The religion and probabilities used match approximate world religion make-up (from Pew Research Center). The default make up is:
<table>
<thead>
<tr>
<th>Religion</th>
<th>N</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Christian</td>
<td>2,173,260,000</td>
<td>31.48%</td>
</tr>
<tr>
<td>Muslim</td>
<td>1,599,280,000</td>
<td>23.16%</td>
</tr>
<tr>
<td>None</td>
<td>1,127,000,000</td>
<td>16.32%</td>
</tr>
<tr>
<td>Hindu</td>
<td>1,034,620,000</td>
<td>14.99%</td>
</tr>
<tr>
<td>Buddhist</td>
<td>489,030,000</td>
<td>7.08%</td>
</tr>
<tr>
<td>Folk</td>
<td>406,140,000</td>
<td>5.88%</td>
</tr>
<tr>
<td>Other</td>
<td>59,330,000</td>
<td>0.86%</td>
</tr>
<tr>
<td>Jewish</td>
<td>15,670,000</td>
<td>0.23%</td>
</tr>
</tbody>
</table>

Value

Returns a random factor vector of religion elements.

References

http://www.pewforum.org/2012/12/18/table-religious-composition-by-country-in-numbers

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

religion(10)
barplot(table(religion(10000)))
pie(table(religion(10000)))

r_data
Pre-Selected Column Data Set

Description

r_data - Generate a data set with pre-set columns selected.

r_data_theme - Generate a themed data set with pre-set columns.

Usage

r_data(n = 500, ...)
r_data_theme(n = 100, data_theme = "the_works")
Arguments

- **n**: The length to pass to the randomly generated vectors (number of rows).
- **data_theme**: A data theme. Currently selections include:
  - *the_works*: all available variable functions
  - *survey*: ID column plus 10 numeric 5-point Likert type response columns
  - *survey2*: ID column plus 10 5-point Likert type response columns
  - ... A set of optionally named arguments. Using *wakefield* variable functions require no name or call parenthesis.

Details

The pre-selected columns include:
- ID
- Race
- Age
- Sex
- Hour
- IQ
- Height
- Died

The user may use ... to add additional columns. *r_data* is a convenience function to quickly produce a data set. For more specific usage use the more flexible *r_data_frame* function.

Value

Returns a tbl_df.

See Also

- *r_data_frame*

Examples

```r
r_data()
r_data(10)
r_data(10, paragraph, Attending = valid)
peek(r_data_theme())
plot(r_data_theme(), flip=TRUE)
r_data_theme(, "survey")
r_data_theme(, "survey2")```
r_data_frame

Data Frame Production (From Variable Functions)

Description
Produce a tbl_df data frame that allows the user to lazily pass unnamed wakefield variable functions (optionally, without call parenthesis).

Usage
r_data_frame(n, ..., rep.sep = "_")

Arguments
n
The length to pass to the randomly generated vectors.
rep.sep
A separator to use for repeated variable names. For example if the age is used three times (r_data_frame(age, age, age)), the name "Age" will be assigned to all three columns. The results in column names c("Age_1", "Age_2", "Age_3"). To turn off this behavior use rep.sep = NULL. This results in c("Age", "Age.1", "Age.2") column names in the data.frame.

... A set of optionally named arguments. Using wakefield variable functions require no name or call parenthesis.

Value
Returns a tbl_df.

Author(s)
Josh O’Brien and Tyler Rinker <tyler.rinker@gmail.com>.

References
http://stackoverflow.com/a/29617983/1000343

See Also
r_list, r_series, r_dummy

Examples
r_data_frame(n = 30,
id, race, age, sex, hour, iq,
```r
r_data_frame

  height,
  died,
  Scoring = rnorm,
  Smoker = valid

r_data_frame(n = 30,
  id,
  race,
  age(x = 8:14),
  Gender = sex,
  Time = hour,
  iq,
  grade, grade, grade, #repeated measures
  height(mean=50, sd = 10),
  died,
  Scoring = rnorm,
  Smoker = valid
)

r_data_frame(n = 500,
  id,
  age, age, age,
  grade, grade, grade
)

## Repeated Measures/Time Series
r_data_frame(n=100,
  id,
  age,
  sex,
  r_series(likert, 3),
  r_series(likert, 4, name = "Item", integer = TRUE)
)

## Expanded Dummy Coded Variables
r_data_frame(n=100,
  id,
  age,
  r_dummy(sex, prefix=TRUE),
  r_dummy(political)
)

## 'peek' to view all columns
## 'plot' ('table_heat') for a graphic representation
library(dplyr)
r_data_frame(n=100,
  id,
  dob,
  animal,
  grade, grade,
  death,
  dummy,
```
r_dummy

Generate Random Dummy Values

Description

Generate random values from a `wakefield` variable function.

Usage

```r
r_dummy(fun, n, ..., prefix = FALSE, rep.sep = "_")
```

Arguments

- `fun` A `wakefield` variable function.
- `n` The number of rows to produce.
- `prefix` logical. If `TRUE` the original factor name (supplied to `fun` as `name` argument) will prefix the column names that were generated from the factor's categories.
- `rep.sep` A separator to use for the variable and category part of names when `prefix = TRUE`. For example if the `age` is used (`r_dummy(sex)`), this results in column names `c("Sex_Male", "Sex_Female")`.
- `...` Additional arguments passed to `fun`.

Value

Returns a `tbl_df`.

See Also

- `r_list`, `r_data_frame`, `r_series`

Examples

```r
r_dummy(sex, 10)
r_dummy(race, 1000)
r_dummy(race, 1000, name = "Ethnicity")```
r_insert Insert Data Frames Into r_data_frame

Description

Safely insert data.frame objects into a r_data_frame or r_list.

Usage

r_insert(x, name = "Inserted")

Arguments

x A data.frame to add a seriesname attribute (i.e., attributes(x)["seriesname"])
name A name to assign to attributes(x)["seriesname"].

Value

Returns a data.frame with a attributes(x)["seriesname"] assigned.

See Also

seriesname

Examples

dat <- dplyr::data_frame(
  Age_1 = age(100), Age_2 = age(100), Age_3 = age(100),
  Smokes = smokes(n=100),
  Sick = ifelse(Smokes, sample(5:10, 100, TRUE), sample(0:4, 100, TRUE)),
  Death = ifelse(Smokes, sample(0:1, 100, TRUE, prob = c(.2, .8)),
                   sample(0:1, 100, TRUE, prob = c(.7, .3)))
)

r_data_frame(10, id,
  r_insert(dat)
)

r_list(10,
  id,
  r_insert(dat)
)
r_list  
List Production (From Variable Functions)

Description

Produce a named list that allows the user to lazily pass unnamed `wakefield` variable functions (optionally, without call parenthesis).

Usage

\[
\text{r_list}(n, \ldots, \text{rep.sep} = "\_")
\]

Arguments

- `n` The length to pass to the randomly generated vectors.
- `rep.sep` A separator to use for repeated variable names. For example, if the `age` is used three times (`r_list(age, age, age)`), the name "Age" will be assigned to all three vectors in the list. The results in column names `c("Age_1", "Age_2", "Age_3")`. To turn off this behavior use `rep.sep = \text{NULL}`. This results in `c("Age", "Age", "Age")` for vector names, leading to `c("Age", "Age.1", "Age.2")` if coerced to a `data.frame`.
- `\ldots` A set of optionally named arguments. Using `wakefield` variable functions require no name or call parenthesis.

Value

Returns a named list of equal length vectors.

Author(s)

Josh O’Brien and Tyler Rinker <tyler.rinker@gmail.com>.

References

http://stackoverflow.com/a/29617983/1000343

See Also

- `r_data_frame`, `r_series`, `r_dummy`

Examples

```r
r_list(n = 30,
      id,
      race,
      age,
      sex,
```
r_na

hour, iq, height, died, Scoring = rnorm
)

r_list(
  n = 30, id, race, age(x = 8:14), Gender = sex, Time = hour, iq, height(mean=50, sd = 10), died, Scoring = rnorm
)

r_na

Replace a Proportion of Values With NA

Description
Replaces a proportion of values with NA. Useful for simulating missing data.

Usage
r_na(x, cols = -1, prob = 0.05)

Arguments
x A data.frame or list to randomly replace elements with NAs.
cols Numeric indices of the columns to include (use - to exclude as well). Default is to assign random NAs to all columns except the first column.
prob The proportion of each column/vector elements to assign to NA.

Value
Returns a data.frame or list with random missing values.

Examples
r_na(mtcars)
r_na(mtcars, NULL)

library(dplyr)
Generate Random Vector

Description
Generate a random vector.

Usage
r_sample(n, x = 1:100, prob = NULL, name = "Sample")

Arguments
n
The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.

x
A vector of elements to chose from.

prob
A vector of probabilities to chose from.

name
The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value
Returns a random vector of elements.

See Also
sample
r_sample_binary

Examples

```r
r_sample(100, name = "Var")
table(r_sample(x = c("Dog", "Cat", "Fish", "Bird"), n=1000))
r_sample(x = c("B", "W"), prob = c(.7, .3), n = 25, name = "Race")
r_sample(25, x = c(TRUE, FALSE))
```

---

### Description

- `r_sample_binary` - Generate a random binary vector.
- `r_sample_binary_factor` - Generate a random binary vector and coerces to a factor.

### Usage

```r
r_sample_binary(n, x = 1:2, prob = NULL, name = "Binary")
r_sample_binary_factor(n, x = 1:2, prob = NULL, name = "Binary")
```

### Arguments

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **x**: A vector of length 2 to sample from.
- **prob**: A vector of probabilities to chose from.
- **name**: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

### Value

Returns a random binary vector of elements.

### See Also

- `sample.int`

### Examples

```r
r_sample_binary(100, name = "Var")
table(r_sample_binary(1000))
c("B", "W")[r_sample_binary(10)]
```
**r_sample_factor**  
*Generate Random Factor Vector*

**Description**
Generate a random vector and coerces to a factor.

**Usage**

```r
r_sample_factor(n, x = LETTERS, prob = NULL, name = "Factor")
```

**Arguments**

- `n` The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x` A vector of elements to chose from.
- `prob` A vector of probabilities to chose from.
- `name` The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Value**
Returns a random actor vector of elements.

**See Also**
`sample`

**Examples**

```r
r_sample_factor(100, name = "Var")
table(r_sample_factor(x = c("Dog", "Cat", "Fish", "Bird"), n=1000))
r_sample_factor(x = c("B", "W"), prob = c(.7, .3), n = 25)
```

---

**r_sample_integer**  
*Generate Random Integer Vector*

**Description**
Generate a random integer vector.

**Usage**

```r
r_sample_integer(n, x = 1:100, prob = NULL, name = "Integer")
```
Arguments

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **x**: A vector of elements to chose from.
- **prob**: A vector of probabilities to chose from.
- **name**: The name to assign to the output vector's `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random integer vector of elements.

See Also

- `sample`

Examples

```r
r_sample_integer(100, name = "Var")
table(r_sample_integer(x = c("Dog", "Cat", "Fish", "Bird"), n=1000))
r_sample_integer(x = c("B", "W"), prob = c(.7, .3), n = 25, name = "Race")
r_sample_integer(25, x = c(TRUE, FALSE))
```

---

Description

Generate Random Logical Vector

Usage

```r
r_sample_logical(n, prob = NULL, name = "Logical")
```

Arguments

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **prob**: A vector of probabilities to chose from.
- **name**: The name to assign to the output vector's `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random logical (TRUE/FALSE) vector of elements.
r_sample_ordered

Generate Random Ordered Factor Vector

Description

Generate a random vector and coerces to an ordered factor.

Usage

```r
r_sample_ordered(n, x = LETTERS[1:5], prob = NULL, name = "Ordered")
```

Arguments

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **x**: A vector of elements to choose from.
- **prob**: A vector of probabilities to choose from.
- **name**: The name to assign to the output vector's `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random factor vector of elements.

See Also

`sample`, `ordered`

Examples

```r
r_sample_ordered(100, name = "Var")
```

```r
lvls <- c("Strongly Agree", "Agree", "Neutral", "Disagree", "Strongly Disagree")
table(r_sample_ordered(x = lvls, n=1000))
```

```r
(out <- r_sample_ordered(x = c("Black", "Grey", "White"),
prob = c(.5, .2, .3), n = 100))
slices <- c(table(out))
pie(slices, main="Pie Chart of Colors", col = tolower(names(slices)))
```
Description

Generate a random vector without replacement.

Usage

\[
\text{r_sample_replace}\text{n, x = 1:100, prob = NULL, replace = FALSE, name = "Sample"}
\]

Arguments

- **n**: The number elements to generate. This can be globally set within the environment of \text{r_data_frame} or \text{r_list}.
- **x**: A vector of elements to chose from.
- **prob**: A vector of probabilities to chose from.
- **replace**: logical. If TRUE sampling is done with replacement. Default is without replacement.
- **name**: The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of \text{r_data_frame} or \text{r_list}.

Value

Returns a random vector of elements.

See Also

\text{sample}

Examples

\[
\text{r_sample}(100, \text{name = "Var"})
\]
\[
\text{table(r_sample(x = c("Dog", "Cat", "Fish", "Bird"), n=1000))}
\]
\[
\text{r_sample(x = c("B", "W"), prob = c(.7, .3), n = 25, name = "Race")}
\]
\[
\text{r_sample}(25, x = c(\text{TRUE, FALSE}))
\]
r_series

Data Frame Series (Repeated Measures)

Description

Produce a tbl_df data frame of repeated measures from a wakefield variable function.

Usage

r_series(fun, j, n, ... , integer = FALSE, relate = NULL, rep.sep = "_")

Arguments

fun
A wakefield variable function.

j
The number of columns to produce.

n
The number of rows to produce.

integer
logical. If TRUE factor columns will be coerced to integer.

relate
Allows the user to specify the relationship between columns. May be a named list of c("operation", "mean", "sd") or a string of the form of "fM_sd" where ‘f’ is one of (+, -, *, /), ‘M’ is a mean value, and ‘sd’ is a standard deviation of the mean value (e.g., "*_4_1"). See relate for details.

rep.sep
A separator to use for repeated variable names. For example if the age is used three times (r_data_frame(age, age, age)), the name "Age" will be assigned to all three columns. The results in column names c("Age_1", "Age_2", "Age_3").

... Additional arguments passed to fun.

Value

Returns a tbl_df.

References

https://github.com/trinker/wakefield/issues/1/#issuecomment-96166910

See Also

r_list, r_data_frame r_dummy

Examples

r_series(grade, 5, 10)

## Custom name prefix
r_series(likert, 5, 10, name = "Question")

## Convert factors to integers
sat

```
r_series(likert_7, 5, 10, integer = TRUE)

## Related variables
r_series(likert, 10, 200, relate = list(operation = "*", mean = 2, sd = 1))
r_series(likert, 10, 200, relate = "--3.1")
r_series(age, 10, 200, relate = "+5.0")

## Change sd to reduce/increase correlation
round(cor(r_series(grade, 10, 10, relate = "+1.2")), 2)
round(cor(r_series(grade, 10, 10, relate = "+1.0")), 2)
round(cor(r_series(grade, 10, 10, relate = "+1.5")), 2)
round(cor(r_series(grade, 10, 10, relate = "+1.20")), 2)

## Plot Example 1
library(dplyr); library(ggplot2)

dat <- r_data_frame(12,
  name,
  r_series(likert, 10, relate = "+1.5")
)

# Suggested use of tidyr or reshape2 package here instead
dat <- data.frame(
  ID = rep(dat[[1]], ncol(dat[-1])),
  stack(dat[-1])
)

dat["Time"] <- factor(sub("Variable_", ",", dat["ind"])), levels = 1:10

ggplot(dat, aes(x = Time, y = values, color = ID, group = ID)) +
  geom_line(size=.8)

## Plot Example 2

dat <- r_data_frame(12,
  name,
  r_series(grade, 100, relate = "+1.2")
)

# Suggested use of tidyr or reshape2 package here instead
dat <- data.frame(
  ID = rep(dat[[1]], ncol(dat[-1])),
  ind = rep(colnames(dat[-1]), each = nrow(dat)),
  values = unlist(dat[-1])
)

dat["Time"] <- as.numeric(sub("Grade_", ",", dat["ind"])))

ggplot(dat, aes(x = Time, y = values, color = ID, group = ID)) +
  geom_line(size=.8) + theme_bw()

---

Generate Random Vector of Scholastic Aptitude Test (SATs)
```
**Description**

```
grade - Generate a random normal vector of scholastic aptitude test (SATs).
```

**Usage**

```
sat(n, mean = 1500, sd = 100, min = 0, max = 2400, digits = 0, name = "SAT")
```

**Arguments**

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **mean**: The mean value for the normal distribution to be drawn from.
- **sd**: The standard deviation of the normal distribution to draw from.
- **min**: A numeric lower boundary cutoff. Results less than this value will be replaced with `min`.
- **max**: A numeric upper boundary cutoff. Results greater than this value will be replaced with `max`.
- **digits**: Integer indicating the number of decimal places to be used. Negative values are allowed (see `round`).
- **name**: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Value**

Returns a random normal vector of SAT elements.

**See Also**

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

**Examples**

```
sat(10)
hist(sat(10000))
interval(sat, 5, n = 1000)
```
second

Generate a Random Sequence of Seconds in H:M:S Format

Description

Generate a random vector of seconds in H:M:S format.

Usage

second(n, x = seq(0, 59, by = 1)/3600, prob = NULL, random = FALSE, name = "Second")

Arguments

n
The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.

x
A vector of elements to chose from.

prob
A vector of probabilities to chose from.

random
logical. If TRUE the times are randomized, otherwise the times are sequential.

name
The name to assign to the output vector’s varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random vector of second time elements in H:M:S format.

See Also

times

Examples

second(20)
second(20, random=TRUE)
pie(table(second(2000, x = seq(0, 59, by = 10)/3600, prob = probs())))
Generate Random Vector of Sentences

Description

Generate a random vector of sentences from the presidential_debates_2012.

Usage

```r
sentence(n, x = wakefield::presidential_debates_2012, prob = NULL,
         name = "Sentence")
```

Arguments

- `n` The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x` A vector of elements to chose from.
- `prob` A vector of probabilities to chose from.
- `name` The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random character vector of sentence elements.

See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

Examples

```r
sentence(10)
```
**seriesname**

*Add Internal Name to Data Frame*

**Description**

Adds attributes(x)["seriesname"] attribute to a data.frame.

**Usage**

`seriesname(x, name)`

**Arguments**

- `x` A data.frame to add a seriesname attribute (i.e., attributes(x)["seriesname"]).
- `name` A name to assign to attributes(x)["seriesname"].

**Value**

Returns a data.frame with a attributes(x)["seriesname"] assigned.

**Examples**

```r
seriesname(mtcars, "Cars")
attributes(seriesname(mtcars, "Cars"))
```

**sex**

*Generate Random Vector of Genders*

**Description**

Generate a random vector of genders.

**Usage**

```r
sex(n, x = c("Male", "Female"), prob = c(0.51219512195122, 0.48780487804878), name = "Sex")

gender(n, x = c("Male", "Female"), prob = c(0.51219512195122, 0.48780487804878), name = "Gender")
```
sex_inclusive

Arguments

- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x`: A vector of length 2 to sample from.
- `prob`: A vector of probabilities to chose from.
- `name`: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Details

The genders and probabilities used match approximate gender make-up:

<table>
<thead>
<tr>
<th>Gender</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>51.22%</td>
</tr>
<tr>
<td>Female</td>
<td>48.78%</td>
</tr>
</tbody>
</table>

Value

Returns a random factor vector of gender elements.

See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

Examples

```r
sex(10)
100*table(sex(n <- 10000))/n
```

Description

Generate a random vector of non-binary genders. Proportion of trans* category was taken from the Williams Institute Report (2011), and subtracted equally from the male and female categories.
sex_inclusive

Usage

sex_inclusive(n, x = c("Male", "Female", "Intersex"), prob = NULL,
name = "Sex")

gender_inclusive(n, x = c("Male", "Female", "Trans*"), prob = NULL,
name = "Gender")

Arguments

n     The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.

x     A vector of elements to chose from.

prob  A vector of probabilities to chose from.

name  The name to assign to the output vector’s varname attribute. This is used to automatically assign names to the column/vector name when used inside of r_data_frame or r_list.

Details

The genders and probabilities used match approximate gender make-up:

<table>
<thead>
<tr>
<th>Gender</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>51.07%</td>
</tr>
<tr>
<td>Female</td>
<td>48.63%</td>
</tr>
<tr>
<td>Trans*</td>
<td>0.30%</td>
</tr>
</tbody>
</table>

Value

Returns a random factor vector of sex or gender elements.

Author(s)

Matthew Sigal <msigal@yorku.ca>

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

sex_inclusive(10)
barplot(table(sex_inclusive(10000)))
gender_inclusive(10)
Generate Random Logical Smokes Vector

**Description**

Generate a random logical (TRUE/FALSE) smokes vector.

**Usage**

```r
smokes(n, prob = c(0.822, 0.178), name = "Smokes")
```

**Arguments**

- **n** The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **prob** A vector of probabilities to chose from.
- **name** The name to assign to the output vector's `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Details**

The probabilities are non-smoker: 82.2% vs. smoker: 17.8%.

**Value**

Returns a random logical vector of smokes elements.

**See Also**

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `speed`, `state`, `string`, `upper`, `valid`, `year`, `zip_code`

**Examples**

```r
smokes(10)
100*table(smokes(n <- 1000))/n
```
speed

Generate Random Vector of Speeds

Description

speed and speed_in - Generate a random normal vector of speeds in inches.

speed_cm - Generate a random normal vector of speeds in centimeters.

Usage

speed(n, mean = 55, sd = 10, min = 0, max = NULL, digits = 0,
      name = "Speed")

speed_mph(n, mean = 55, sd = 10, min = 0, max = NULL, digits = 1,
          name = "Speed(mph)")

speed_kph(n, mean = 88.5, sd = 16, min = 0, max = NULL, digits = 1,
          name = "Speed(kph)")

Arguments

n The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.

mean The mean value for the normal distribution to be drawn from.

sd The standard deviation of the normal distribution to draw from.

min A numeric lower boundary cutoff. Results less than this value will be replaced with min.

max A numeric upper boundary cutoff. Results greater than this value will be replaced with max.

digits Integer indicating the number of decimal places to be used. Negative values are allowed (see round).

name The name to assign to the output vector's varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random normal vector of speed elements.

Note

speed rounds to nearest whole number. speed_in & speed_in round to the nearest tenths.
See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, state, string, upper, valid, year, zip_code

Examples

```r
speed(10)
hist(speed(10000))
interval(speed, 5, n = 1000)
```

---

**state**

`Generate Random Vector of states`

**Description**

Generate a random factor vector of states.

**Usage**

```r
state(n, x = datasets::state.name,
     prob = wakefield::state_populations[["Proportion"]], name = "State")
```

**Arguments**

- `n` The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x` A vector of elements to chose from.
- `prob` A vector of probabilities to chose from.
- `name` The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Details**

The state populations and probabilities:

<table>
<thead>
<tr>
<th>State</th>
<th>Population</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>California</td>
<td>37,253,956</td>
<td>12.09 %</td>
</tr>
<tr>
<td>Texas</td>
<td>25,145,561</td>
<td>8.16 %</td>
</tr>
<tr>
<td>New York</td>
<td>19,378,102</td>
<td>6.29 %</td>
</tr>
<tr>
<td>Florida</td>
<td>18,801,310</td>
<td>6.10 %</td>
</tr>
<tr>
<td>Illinois</td>
<td>12,830,632</td>
<td>4.16 %</td>
</tr>
<tr>
<td>Pennsylvania</td>
<td>12,702,379</td>
<td>4.12 %</td>
</tr>
<tr>
<td>State</td>
<td>Population</td>
<td>Percentage</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>Ohio</td>
<td>11,536,504</td>
<td>3.74%</td>
</tr>
<tr>
<td>Michigan</td>
<td>9,883,640</td>
<td>3.21%</td>
</tr>
<tr>
<td>Georgia</td>
<td>9,687,653</td>
<td>3.14%</td>
</tr>
<tr>
<td>North Carolina</td>
<td>9,535,483</td>
<td>3.09%</td>
</tr>
<tr>
<td>New Jersey</td>
<td>8,791,894</td>
<td>2.85%</td>
</tr>
<tr>
<td>Virginia</td>
<td>8,001,024</td>
<td>2.60%</td>
</tr>
<tr>
<td>Washington</td>
<td>6,724,540</td>
<td>2.18%</td>
</tr>
<tr>
<td>Massachusetts</td>
<td>6,547,629</td>
<td>2.12%</td>
</tr>
<tr>
<td>Indiana</td>
<td>6,483,802</td>
<td>2.10%</td>
</tr>
<tr>
<td>Arizona</td>
<td>6,392,017</td>
<td>2.07%</td>
</tr>
<tr>
<td>Tennessee</td>
<td>6,346,105</td>
<td>2.06%</td>
</tr>
<tr>
<td>Missouri</td>
<td>5,988,927</td>
<td>1.94%</td>
</tr>
<tr>
<td>Maryland</td>
<td>5,773,552</td>
<td>1.87%</td>
</tr>
<tr>
<td>Wisconsin</td>
<td>5,686,986</td>
<td>1.85%</td>
</tr>
<tr>
<td>Minnesota</td>
<td>5,303,925</td>
<td>1.72%</td>
</tr>
<tr>
<td>Colorado</td>
<td>5,029,196</td>
<td>1.63%</td>
</tr>
<tr>
<td>Alabama</td>
<td>4,779,736</td>
<td>1.55%</td>
</tr>
<tr>
<td>South Carolina</td>
<td>4,625,364</td>
<td>1.50%</td>
</tr>
<tr>
<td>Louisiana</td>
<td>4,533,372</td>
<td>1.47%</td>
</tr>
<tr>
<td>Kentucky</td>
<td>4,339,367</td>
<td>1.41%</td>
</tr>
<tr>
<td>Oregon</td>
<td>3,831,074</td>
<td>1.24%</td>
</tr>
<tr>
<td>Oklahoma</td>
<td>3,751,351</td>
<td>1.22%</td>
</tr>
<tr>
<td>Connecticut</td>
<td>3,574,097</td>
<td>1.16%</td>
</tr>
<tr>
<td>Iowa</td>
<td>3,046,355</td>
<td>.99%</td>
</tr>
<tr>
<td>Mississippi</td>
<td>2,967,297</td>
<td>.96%</td>
</tr>
<tr>
<td>Arkansas</td>
<td>2,915,918</td>
<td>.95%</td>
</tr>
<tr>
<td>Kansas</td>
<td>2,853,118</td>
<td>.93%</td>
</tr>
<tr>
<td>Utah</td>
<td>2,763,885</td>
<td>.90%</td>
</tr>
<tr>
<td>Nevada</td>
<td>2,700,551</td>
<td>.88%</td>
</tr>
<tr>
<td>New Mexico</td>
<td>2,059,179</td>
<td>.67%</td>
</tr>
<tr>
<td>West Virginia</td>
<td>1,852,994</td>
<td>.60%</td>
</tr>
<tr>
<td>Nebraska</td>
<td>1,826,341</td>
<td>.59%</td>
</tr>
<tr>
<td>Idaho</td>
<td>1,567,582</td>
<td>.51%</td>
</tr>
<tr>
<td>Hawaii</td>
<td>1,360,301</td>
<td>.44%</td>
</tr>
<tr>
<td>Maine</td>
<td>1,328,361</td>
<td>.43%</td>
</tr>
<tr>
<td>New Hampshire</td>
<td>1,316,470</td>
<td>.43%</td>
</tr>
<tr>
<td>Rhode Island</td>
<td>1,052,567</td>
<td>.34%</td>
</tr>
<tr>
<td>Montana</td>
<td>989,415</td>
<td>.32%</td>
</tr>
<tr>
<td>Delaware</td>
<td>897,934</td>
<td>.29%</td>
</tr>
<tr>
<td>South Dakota</td>
<td>814,180</td>
<td>.26%</td>
</tr>
<tr>
<td>Alaska</td>
<td>710,231</td>
<td>.23%</td>
</tr>
<tr>
<td>North Dakota</td>
<td>672,591</td>
<td>.22%</td>
</tr>
<tr>
<td>Vermont</td>
<td>625,741</td>
<td>.20%</td>
</tr>
<tr>
<td>Wyoming</td>
<td>563,626</td>
<td>.18%</td>
</tr>
</tbody>
</table>

**Value**

Returns a random character vector of state elements.
See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, string, upper, valid, year, zip_code

Examples

    state(10)
    pie(table(state(10000)))
    sort(100*table(state(n <- 10000))/n)

state_populations  State Populations (2010)

Description

A dataset containing U.S. state populations.

Usage

    data(state_populations)

Format

A data frame with 50 rows and 3 variables

Details

• State. The 50 U.S. states.
• Population. Population of state.
• Proportion. Proportion of total U.S. population.

References

Description

Generate a random vector of strings.

Usage

```r
string(n, x = "[A-Za-z0-9]", length = 10, name = "String")
```

Arguments

- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x`: A character vector specifying character classes to draw elements from.
- `length`: Integer vector, desired string lengths.
- `name`: The name to assign to the output vector’s `varname` attribute. This is used to automatically assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random character vector of string elements.

See Also

`stri_rand_strings`

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `upper`, `valid`, `year`, `zip_code`

Examples

```r
string(10)
```
table_heat  

View Data Table Column Types as Heat Map

Description

Generate a heat map of column types from a data.frame.

Usage

```r
table_heat(x, flip = FALSE, palette = "Set3", print = interactive(), sep = "\n")
```

Arguments

- `x`: A data.frame.
- `flip`: logical. If TRUE the data.frame is flipped so that the columns are on the y axis and observations on the x axis. This is useful when there are many columns or the column names are longer.
- `palette`: A palette to chose from. See `scale_fill_brewer` for more. These choices should exceed the number of unique column types. Use NULL to use ggplot2’s default color scheme.
- `print`: logical. If TRUE the plot is printed. Option for use in document construction such as knitr or rmarkdown.
- `sep`: A separator to use between column types. Column types are determined via `sapply(x, class)`. When multiple types are present these are collapsed. By default the \n is used.

Details

By default column names retain their order. Column types are ordered alphabetically in the legend, with NA appearing last.

Value

Returns a ggplot2 object.

Examples

```r
table_heat(mtcars) #boring
table_heat(C02)
table_heat(iris)
table_heat(state_populations)

dat <- r_data_frame(100, 
               lorem_ipsum, 
               birth, 
               animal,
)
Generate a Random Sequence of Times in H:M:S Format

**Description**

Generate a random vector of times in H:M:S format.

**Usage**

```r
time_stamp(n, x = seq(0, 23, by = 1), prob = NULL, random = FALSE, name = "Time")
```

**Arguments**

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **x**: A vector of elements to chose from.
- **prob**: A vector of probabilities to chose from.
- **random**: Logical. If `TRUE` the times are randomized, otherwise the times are sequential.
- **name**: The name to assign to the output vector’s `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

```r
table_heat(dat)
table_heat(dat, flip=TRUE)

## NA values
table_heat(r_na(dat, NULL))

## Colors
table_heat(r_na(dat, NULL), palette = "null")
```
Value

Returns a random vector of time elements in H:M:S format.

See Also
times

Examples

time_stamp(20)
time_stamp(20, random=TRUE)
pie(table(time_stamp(2000, x = seq(0, 23, by = 2), prob = probs(12))))

Generate Random Letter Vector

Description

upper - Generates a random character vector of upper case letters.
lower - Generates a random character vector of lower case letters.
upper_factor - Generates a random factor vector of upper case letters.
lower_factor - Generates a random factor vector of lower case letters.

Usage

upper(n, k = 5, x = LETTERS, prob = NULL, name = "Upper")
lower(n, k = 5, x = c("a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z"),
prob = NULL, name = "Lower")
upper_factor(n, k = 5, x = LETTERS, prob = NULL, name = "Upper")
lower_factor(n, k = 5, x = c("a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z"),
prob = NULL, name = "Lower")

Arguments

n The number elements to generate. This can be globally set within the environ-
ment of r_data_frame or r_list.
k The number of the elements of x to sample from (uses 1:k).
x A vector of elements to chose from.
prob A vector of probabilities to chose from.
name The name to assign to the output vector’s varname attribute. This is used to auto
assign names to the column/vector name when used inside of r_data_frame or
r_list.
valid

Value

Returns a random character/factor vector of letter elements.

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, valid, year, zip_code

Examples

upper(10)
lower(10)
upper_factor(10)
lower_factor(10)
barplot(table(upper(10000)))
barplot(table(upper(10000, prob = probs(5))))

valid

Generate Random Logical Vector

Description

Generate a random logical (TRUE/FALSE) vector.

Usage

valid(n, prob = NULL, name = "Valid")

Arguments

n

The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.

prob

A vector of probabilities to chose from.

name

The name to assign to the output vector’s var_name attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random logical vector of elements.
See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, year, zip_code

Examples

valid(10)
100*table(valid(n <- 1000))/n

<table>
<thead>
<tr>
<th>variables</th>
<th>Available Variable Functions</th>
</tr>
</thead>
</table>

Description

See a listing of all available variable functions for use in `r_data_frame` or `r_list`.

Usage

`variables(type = NULL, ncols = 5, ...)`

Arguments

- `type`: The output type. Must be either `NULL` (returns a character vector), "matrix", or "list"; or the user may extract a specific type from a list using: "character", "date", "factor", "integer", "logical", "numeric", "ordered factor". Setting `type = TRUE` will also return a list. The `list` version breaks the variable functions into classes. Specifying a specific class (e.g., `type = "numeric"`) will list only variable functions that yield a numeric output.
- `ncols`: The number of columns to use if `type = "matrix"`.
- `...`: Other arguments passed to `matrix`.

Value

Returns a `character` vector, `matrix` of all variable functions, or a `list` of variable functions by type.

Examples

`variables()`

`variables("list")`
`variables(TRUE)`
`names(variables("list"))`
`variables("ordered factor")`
`variables("numeric")`
Add Internal Name to Vector

Description

Adds the class variable and an internal attributes(x)["varname"] attribute to a vector.

Usage

```
varname(x, name)
```

Arguments

- `x` A vector to add a `varname` attribute (i.e., `attributes(x)["varname"]`).
- `name` A name to assign to `attributes(x)["varname"]`.

Value

Returns a vector of the class variable with a `attributes(x)["varname"]` assigned.

Examples

```
varname(1:10, "A")
attributes(varname(1:10, "A"))
sum(varname(1:10, "A"))

varname(LETTERS, "Caps")
attributes(varname(LETTERS, "Caps"))
paste(varname(LETTERS, "Caps"), collapse="")
```

Generate Random Data Sets

Description

Generates random data sets including: data.frames, lists, and vectors.
Generate Random Vector of Years

Description

Generate a random vector of years.

Usage

```r
year(n, x = 1996:as.numeric(format(Sys.Date(), "%Y")), prob = NULL,
   name = "Year")
```

Arguments

- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `x`: A vector of elements to chose from.
- `prob`: A vector of probabilities to chose from.
- `name`: The name to assign to the output vector's `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Value

Returns a random vector of year elements.

See Also

Other variable functions: `age`, `animal`, `answer`, `area`, `car`, `children`, `coin`, `color`, `date_stamp`, `death`, `dice`, `dna`, `dob`, `dummy`, `education`, `employment`, `eye`, `grade_level`, `grade`, `group`, `hair`, `height`, `income`, `internet_browser`, `iq`, `language`, `level`, `likert`, `lorem_ipsum`, `marital`, `military`, `month`, `name`, `normal`, `political`, `race`, `religion`, `sat`, `sentence`, `sex_inclusive`, `sex`, `smokes`, `speed`, `state`, `string`, `upper`, `valid`, `zip_code`

Examples

```r
year(10)
pr <- probs(length(1996:2016))
pie(table(year(10000, x = 1996:2016, prob = pr)))
```
Describe the vector of zip codes.

Usage

zip_code(n, k = 10, x = 10000:99999, prob = NULL, name = "Zip")

Arguments

n The number of elements to generate. This can be globally set within the environment of r_data_frame or r_list.

k The number of the elements of x to sample from (uses sample(x, k)).

x A vector of elements to choose from.

prob A vector of probabilities to choose from.

name The name to assign to the output vector's varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Returns a random vector of zip code elements.

See Also

Other variable functions: age, animal, answer, area, car, children, coin, color, date_stamp, death, dice, dna, dob, dummy, education, employment, eye, grade_level, grade, group, hair, height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year

Examples

zip_code(10)
pie(table(zip_code(10000, prob = probs(10))))
Index

*Topic **age**  
  age, 4
*Topic **animal**  
  animal, 5
*Topic **answer**  
  answer, 6
*Topic **area**  
  area, 7
*Topic **army**  
  military, 41
*Topic **birth**  
  dob, 17
*Topic **branch**  
  military, 41
*Topic **browser**  
  internet_browser, 32
*Topic **capitals**  
  upper, 84
*Topic **car**  
  car, 9
*Topic **character**  
  lorem_ipsum, 39
  string, 81
*Topic **children**  
  children, 10
*Topic **class**,  
  table_heat, 82
*Topic **coin**  
  coin, 11
*Topic **color**  
  color, 12
*Topic **correlate**  
  relate, 52
*Topic **cut**  
  interval, 33
*Topic **datasets**  
  animal_list, 6
  grady_augmented, 25
  languages, 36
  name_neutral, 45
  presidential.debates.2012, 49
  state_populations, 80
*Topic **date**  
  date_stamp, 13
*Topic **death**  
  death, 14
*Topic **democrat**  
  political, 48
*Topic **dice**  
  dice, 15
*Topic **died**  
  death, 14
*Topic **divorce**  
  marital, 40
*Topic **dna**  
  dna, 16
*Topic **dob**  
  dob, 17
*Topic **dummy**  
  r_dummy, 58
*Topic **education**  
  education, 19
*Topic **employment**  
  employment, 20
*Topic **eye**  
  eye, 21
*Topic **factor**  
  r_sample_ordered, 66
*Topic **false**  
  valid, 85
*Topic **gender**  
  sex, 73
  sex_inclusive, 74
*Topic **gpa**  
  grade, 22
*Topic **grade**  
  grade, 22
  grade_level, 24
*Topic group
group, 25

*Topic hair
hair, 26

*Topic head
coin, 11

*Topic height
height, 27

*Topic hour
hour, 29

*Topic identification
id, 30

*Topic id
id, 30

*Topic income
income, 31

*Topic insert
r_insert, 59

*Topic integer
as_integer, 8

*Topic intelligence
iq, 34

*Topic interval
interval, 33

*Topic iq
iq, 34

*Topic iris
eye, 21

*Topic language
language, 35

*Topic letters
upper, 84

*Topic level
level, 37

*Topic likert
likert, 38

*Topic list
r_list, 60

*Topic logical
valid, 85

*Topic lower
upper, 84

*Topic marines
military, 41

*Topic marital
marital, 40

*Topic married
marital, 40

*Topic military
military, 41

*Topic minute
minute, 42

*Topic missing
r_na, 61

*Topic month
month, 43

*Topic name
name, 44

*Topic navy
military, 41

*Topic na
r_na, 61

*Topic normal
normal, 45

*Topic no
answer, 6

*Topic numeric
as_integer, 8

*Topic ordered
r_sample_ordered, 66

*Topic percent
probs, 50

*Topic pet
animal, 5

*Topic political
political, 48

*Topic politics
political, 48

*Topic probability
probs, 50

*Topic race
race, 51

*Topic related
relate, 52

*Topic religion
religion, 53

*Topic republican
political, 48

*Topic responses
likert, 38

*Topic sat
sat, 69

*Topic second
second, 71

*Topic sentence
sentence, 72
*Topic sex
  sex, 73
  sex_inclusive, 74
*Topic smoking
  smokes, 76
*Topic speed
  speed, 77
*Topic state
  state, 78
*Topic string
  lorem_ipsum, 39
  string, 81
*Topic tail
  coin, 11
*Topic time
  hour, 29
  minute, 42
  second, 71
  time_stamp, 83
*Topic true
  valid, 85
*Topic type
  table_heat, 82
  variables, 86
*Topic upper
  upper, 84
*Topic valid
  valid, 85
*Topic widow
  marital, 40
*Topic year
  year, 88
*Topic yes
  answer, 6
*Topic zip_code
  zip_code, 89

area, 4, 5, 7, 7, 9–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
as.integer, 8
as_integer, 8

birth (dob), 17
car, 4, 5, 7, 8, 9, 10–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
color, 4, 5, 7–11, 12, 13–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
cut, 33, 34
data.frame, 8, 46, 56, 59–61, 73, 82
date_stamp, 4, 5, 7–12, 13, 14–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
dead, 4, 5, 7–13, 14, 15–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
dice, 4, 5, 7–14, 15, 16–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
died (death), 14
dna, 4, 5, 7–15, 16, 17, 18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
dob, 4, 5, 7–16, 17, 18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
dummy, 4, 5, 7–17, 18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
education, 4, 5, 7–18, 19, 21–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
plot.tbl_df, 47
political, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 48, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
presidential_debates_2012, 35, 49, 72
primary (color), 12
print-available, 49
print-variable, 50
probs, 50
r_data, 54
r_data-frame, 55, 56, 58–60, 68, 86
r_data-theme (r_data), 54
r_dummy, 56, 58, 60, 68
r_insert, 59
r_list, 56, 58, 59, 60, 68, 86
r_na, 61
r_sample, 62
r_sample-binary, 63
r_sample-binary-factor
  (r_sample-binary), 63
r_sample_factor, 64
r_sample_integer, 64
r_sample_logical, 65
r_sample_ordered, 66
r_sample_replace, 67
r_series, 8, 52, 56, 58, 60, 68
race, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
relate, 52, 68
religion, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 53, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
rnorm, 45, 46
round, 23, 28, 31, 34, 46, 70, 77
sample, 62, 64–67
sample.int, 18, 63
sat, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 69, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
scale_fill_brewer, 82
second, 71
sentence, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
seq.Date, 13, 17
seriesname, 59, 73
sex, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 73, 75, 76, 78, 80, 81, 85, 86, 88, 89
sex_inclusive, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74, 76, 78, 80, 81, 85, 86, 88, 89
smokes, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74, 75, 76, 78, 80, 81, 85, 86, 88, 89
speed, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 77, 80, 81, 85, 86, 88, 89
speed_kph(speed), 77
speed_mph(speed), 77
sprintf, 30
state, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
state_populations, 80
stri_rand_lipsum, 39, 40
stri_rand_strings, 81
string, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
table_heat, 48, 82
tbl_df, 52, 55, 56, 58, 68
time_stamp, 83
times, 29, 42, 71, 84
upper, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 84, 86, 88, 89
upper_factor (upper), 84
valid, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
variables, 86
varname, 87
wakefield, 87
wakefield-package (wakefield), 87
year, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 83, 86, 88, 89
zip_code, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 83, 86, 88, 89