Package 'baizer'

January 10, 2023

Title Useful Functions for Data Processing

Version 0.1.0

Description

In ancient Chinese mythology, Bai Ze is a divine creature that knows the needs of everything. 'baizer' provides data processing functions frequently used by the author. Hope this package also knows what you want!

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

Imports dplyr, magrittr, purrr, rlang, stats, stringr, tibble

Suggests covr, testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 2.10)

LazyData true

URL https://github.com/william-swl/baizer

BugReports https://github.com/william-swl/baizer/issues

NeedsCompilation no

Author William Song [aut, cre]

Maintainer William Song <william_swl@163.com>

Repository CRAN

Date/Publication 2023-01-10 18:40:08 UTC

R topics documented:

`c2r` *wrapper of tibble::column_to_rownames*

Description

wrapper of tibble::column_to_rownames

Usage

```
c2r(df, col = "")
```

Arguments

`df`	tibble
`col`	a col name

Value

data.frame

Examples

```
mini_diamond %>% c2r("id")
```

collapse_vector *dump a named vector into character*

Description

dump a named vector into character

Usage

```
collapse_vector(named_vector, front_name = TRUE, collapse = ",")
```

Arguments

named_vector	a named vector
front_name	if TRUE, put names to former
collapse	collapse separator

Value

character

Examples

```
collapse_vector(c(e = 1:4), front_name = TRUE, collapse = ";")
```

detect_dup *detect possible duplication in a vector, ignore case, blank and special character*

Description

detect possible duplication in a vector, ignore case, blank and special character

Usage

```
detect_dup(vector, index = FALSE)
```

Arguments

vector	vector possibly with duplication
index	return duplication index

Value

duplication sub-vector

Examples

```
detect_dup(c("a", "C_", "c -", "#A"))
```

---

diff_index *the index of nth different character*

Description

the index of nth different character

Usage

```
diff_index(s1, s2, nth = 0)
```

Arguments

s1	string1
s2	string2
nth	return the index of nth different character. if 0 return all the indices

Value

the index of differences

Examples

```
diff_index("ATTC", "ATAC")
```

---

fancy_count *better count to show a main column and a fine column*

Description

better count to show a main column and a fine column

Usage

```
fancy_count(df, main_group, fine_group, fine_fmt = "count", sort = TRUE)
```

Arguments

df	tibble
main_group	main column as main group
fine_group	fine column as subgroup
fine_fmt	output fine column format, count|ratio|clean
sort	sort by frequency or not

Value

tibble

Examples

```
fancy_count(mini_diamond, "cut", "clarity")
```

---

fix_to_regex *trans fixed string into regular expression string*

Description

trans fixed string into regular expression string

Usage

```
fix_to_regex(p)
```

Arguments

p	raw fixed pattern

Value

regex pattern

Examples

```
fix_to_regex("ABC|?(*)")
```

---

float_to_percent *from float number to percent number*

Description

from float number to percent number

Usage

```
float_to_percent(x, digits = 2)
```

Arguments

x	number
digits	hold n digits after the decimal point

Value

percent character of x

Examples

```
float_to_percent(0.12)
```

`is.zero` *if a number only have zeros*

Description

if a number only have zeros

Usage

```
is.zero(x)
```

Arguments

x	number

Value

all zero or not

Examples

```
is.zero("0.00")
```

`mini_diamond` *Minimal tibble dataset adjusted from diamond*

Description

Minimal tibble dataset adjusted from diamond

Usage

```
mini_diamond
```

Format

`mini_diamond`**:**

A data frame with 100 rows and 7 columns:

id unique id

cut, clarity 2 category variables

carat, price, x, y 4 continuous variables ...

Source

adjusted from `ggplot2`

`number_fun_wrapper` *wrapper of the functions to process number string with prefix and suffix*

Description

wrapper of the functions to process number string with prefix and suffix

Usage

```
number_fun_wrapper(
  x,
  fun = ~.x,
  prefix_ext = NULL,
  suffix_ext = NULL,
  verbose = FALSE
)
```

Arguments

`x`	number string vector with prefix and suffix
`fun`	process function
`prefix_ext`	prefix extension
`suffix_ext`	suffix extension
`verbose`	print more details

Value

processed number with prefix and suffix

Examples

```
number_fun_wrapper(">=2.134%", function(x) round(x, 2))
```

ordered_slice *better slice by an ordered vector*

Description

better slice by an ordered vector

Usage

```
ordered_slice(df, by, ordered_vector, na.rm = FALSE, dup.rm = FALSE)
```

Arguments

df	tibble
by	slice by this column, this value must has no duplicated value
ordered_vector	ordered vector
na.rm	remove NA or unknown values from ordered vector
dup.rm	remove duplication values from ordered vector

Value

sliced tibble

Examples

```
ordered_slice(mini_diamond, "id", c("id-3", "id-2"))
```

percent_to_float *from percent number to float number*

Description

from percent number to float number

Usage

```
percent_to_float(x, digits = 2)
```

Arguments

x	percent number character
digits	hold n digits after the decimal point

Value

float character of x

Examples

```
percent_to_float("12%")
```

---

r2c *wrapper of tibble::rownames_to_column*

Description

wrapper of tibble::rownames_to_column

Usage

```
r2c(df, col = "")
```

Arguments

df	tibble
col	a col name

Value

tibble

Examples

```
mini_diamond %>%
  c2r("id") %>%
  r2c("id")
```

---

round_string *from float number to fixed digits character*

Description

from float number to fixed digits character

Usage

```
round_string(x, digits = 2)
```

Arguments

x	number
digits	hold n digits after the decimal point

Value

character

Examples

```
round_string(1.1, 2)
```

---

signif_round_string *signif or round strings*

Description

signif or round strings

Usage

```
signif_round_string(x, digits = 2, format = "short")
```

Arguments

x	number
digits	signif or round digits
format	short or long

Value

signif or round strings

Examples

```
signif_round_string(0.03851)
```

signif_string *from float number to fixed significant digits character*

Description

from float number to fixed significant digits character

Usage

```
signif_string(x, digits = 2)
```

Arguments

x	number
digits	hold n significant digits

Value

character

Examples

```
signif_string(1.1, 2)
```

%neq% *not equal calculation operator, support NA*

Description

not equal calculation operator, support NA

Usage

```
x %neq% y
```

Arguments

x	value x
y	value y

Value

logical value, TRUE if x and y are not equal

Examples

```
1 %neq% NA
```

`%nin%` *not in calculation operator*

Description

not in calculation operator

Usage

```
left %nin% right
```

Arguments

`left`	left element
`right`	right element

Value

logical value, TRUE if left is not in right

Examples

```
0 %nin% 1:4
```

Index