Package ‘name’

October 13, 2022

Title Tools for Working with Names

Version 0.0.1

Description A system for organizing column names in data. Aimed at supporting a prefix-based and suffix-based column naming scheme. Extends 'dplyr' functionality to add ordering by function and more explicit renaming.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.2.0

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Imports tibble, dplyr, stringr, tidyselect, rlang, cli

URL `https://github.com/christopherkenny/name`,
`https://christophertkenny.com/name/`

BugReports `https://github.com/christopherkenny/name/issues`

Depends R (>= 4.1)

NeedsCompilation no

Author Christopher T. Kenny [aut, cre] (<`https://orcid.org/0000-0002-9386-6860`>)

Maintainer Christopher T. Kenny `<christopherkenny@fas.harvard.edu>`

Repository CRAN

Date/Publication 2022-08-11 15:00:02 UTC

R topics documented:

`add_pref` *Add Prefix*

Description

Add Prefix

Usage

```
add_pref(x, pref)
```

Arguments

`x`	character; string to change
`pref`	character; prefix to add

Value

character

Examples

```
x <- c('pop', 'pop_2020_est', 'pop_white_2020', 'pop_black_2020')
add_pref(x, 'census_')
```

`add_suff` *Add Suffix*

Description

Add Suffix

Usage

```
add_suff(x, suff)
```

Arguments

x	character; string to change
suff	character; suffix to add

Value

character

Examples

```
x <- c('pop', 'pop_2020_est', 'pop_white_2020', 'pop_black_2020')
add_suff(x, '_cen')
```

`compare_names` *Compare the Names of Two Objects*

Description

Compare the Names of Two Objects

Usage

```
compare_names(x, y)
```

Arguments

x	first object
y	second object

Value

character vector of differences, invisibly

Examples

```
s <- tibble::tibble(a = 1, b = 2, d = 3)
t <- tibble::tibble(a = 1, d = 3, c = 2)
compare_names(s, t)
```

list_phrase *List Phrases*

Description

List Phrases

Usage

```
list_phrase(tb, loc = 2)
```

Arguments

tb	tibble; data to list prefixes in
loc	number of location to list. For example 1_2_3_4 with loc = 3 returns 3.

Value

character

Examples

```
tb <- tibble::tibble(pop = 10, pop_2020_est = 9, pop_white_2020 = 8, pop_black_2020 = 2)
list_phrase(tb)
```

list_pref *List Prefixes*

Description

List Prefixes

Usage

```
list_pref(tb)
```

Arguments

tb	tibble; data to list prefixes in

Value

character

Examples

```
tb <- tibble::tibble(pop = 10, pop_2020_est = 9, pop_white_2020 = 8, pop_black_2020 = 2)
list_pref(tb)
```

`list_suff` *List Suffixes*

Description

List Suffixes

Usage

```
list_suff(tb)
```

Arguments

`tb`	tibble; data to list prefixes in

Value

character

Examples

```
tb <- tibble::tibble(pop = 10, pop_2020_est = 9, pop_white_2020 = 8, pop_black_2020 = 2)
list_suff(tb)
```

`relocate_with` *Relocate columns*

Description

Relocate columns

Usage

```
relocate_with(
  .data,
  .fn,
  .cols = everything(),
  .before = NULL,
  .after = NULL,
  ...
)
```

Arguments

`.data` A `data.frame` or `tibble`.

`.fn` A function to reorder `.cols`.

`.cols` Columns to move

`.before, .after` Destination of columns. If both selected, errors. If neither, moves to right of first selected column.

`...` additional arguments to pass to `.fn`

Value

And object with same type as `.data`.

Examples

```
data(sd)
sd |> relocate_with(sort)
```

`rem_phrase` *Remove Phrase*

Description

Remove Phrase

Usage

```
rem_phrase(x, phrase)
```

Arguments

`x` character; string to change

`phrase` character; phrase to remove

Value

character

Examples

```
x <- c('pop', 'pop_2020_est', 'pop_white_2020', 'pop_black_2020')
rem_phrase(x, '_2020')
```

`rem_pref` *Remove Prefix*

Description

Remove Prefix

Usage

```
rem_pref(x, pref)
```

Arguments

`x`	character; string to change
`pref`	character; prefix to remove

Value

character

Examples

```
x <- c('pop', 'pop_2020_est', 'pop_white_2020', 'pop_black_2020')
rem_pref(x, 'pop_')
```

`rem_suff` *Remove Suffix*

Description

Remove Suffix

Usage

```
rem_suff(x, suff)
```

Arguments

`x`	character; string to change
`suff`	character; suffix to remove

Value

character

Examples

```
x <- c('pop', 'pop_2020_est', 'pop_white_2020', 'pop_black_2020')
rem_suff(x, '_2020')
```

`rename_with_loud` *Rename with, but Loudly*

Description

Rename with, but Loudly

Usage

```
rename_with_loud(.data, .fn, .cols = everything(), ...)
```

Arguments

`.data` A data frame, data frame extension (e.g. a tibble), or a lazy data frame (e.g. from dbplyr or dtplyr). See *Methods*, below, for more details.

`.fn` A function used to transform the selected `.cols`. Should return a character vector the same length as the input.

`.cols` <`tidy-select`> Columns to rename; defaults to all columns.

`...` For `rename()`: <`tidy-select`> Use `new_name` = `old_name` to rename selected variables.

For `rename_with()`: additional arguments passed onto `.fn`.

Value

.data renamed

Examples

```
tb <- tibble::tibble(pop = 10, pop_2020_est = 9, pop_white_2020 = 8, pop_black_2020 = 2)
rename_with_loud(tb, \(x) rem_suff(x, '_2020'))
```

repl_phrase *Replace Phrase*

Description

Replace Phrase

Usage

```
repl_phrase(x, phrase, repl)
```

Arguments

x	character; string to change
phrase	character; phrase to replace
repl	character; phrase to replace with

Value

character

Examples

```
x <- c('pop', 'pop_2020_est', 'pop_white_2020', 'pop_black_2020')
repl_phrase(x, '_2020', '_20')
```

repl_pref *Replace Prefix*

Description

Replace Prefix

Usage

```
repl_pref(x, pref, repl)
```

Arguments

x	character; string to change
pref	character; prefix to replace
repl	character; prefix to replace with

Value

character

Examples

```
x <- c('pop', 'pop_2020_est', 'pop_white_2020', 'pop_black_2020')
repl_pref(x, 'pop_', 'p_')
```

`repl_suff` *Replace Suffix*

Description

Replace Suffix

Usage

```
repl_suff(x, suff, repl)
```

Arguments

`x`	character; string to change
`suff`	character; suffix to replace
`repl`	character; suffix to replace with

Value

character

Examples

```
x <- c('pop', 'pop_2020_est', 'pop_white_2020', 'pop_black_2020')
repl_suff(x, '_2020', '_20')
```

`sd` *South Dakota Election and Demographic Data*

Description

This data set contains demographic and election information for South Dakota

Usage

```
data("sd")
```

References

Voting and Election Science Team, 2020, "2020 Precinct-Level Election Results", https://doi.org/10.7910/DVN/K7760H, Harvard Dataverse, V23

Voting and Election Science Team, 2018, "2016 Precinct-Level Election Results", https://doi.org/10.7910/DVN/NH5S2I, Harvard Dataverse, V71

Voting and Election Science Team, 2019, "2018 Precinct-Level Election Results", https://doi.org/10.7910/DVN/UBKYRU, Harvard Dataverse, V48

Kenny & McCartan (2021, Aug. 10). ALARM Project: 2020 Redistricting Data Files. Retrieved from `https://github.com/alarm-redist/census-2020/`

Examples

```
data(sd)
```

`sort_phrase` *Sort by Phrase*

Description

Sort by Phrase

Usage

```
sort_phrase(x, loc = 2)
```

Arguments

`x` character; strings to sort

`loc` number of location to sort by. For example `1_2_3_4` with `loc = 3` sorts by 3.

Value

character

Examples

```
x <- c('pop_2020_est', 'pop_white_2020', 'pop_black_2020', 'pop_white_2021')
sort_phrase(x)
```

sort_pref *Sort by Prefix*

Description

Sort by Prefix

Usage

```
sort_pref(x)
```

Arguments

x	character; strings to sort

Value

character

Examples

```
x <- c('pop', 'pop_2020_est', 'pop_white_2020', 'pop_black_2020')
sort_pref(x)
```

sort_suff *Sort by Suffix*

Description

Sort by Suffix

Usage

```
sort_suff(x)
```

Arguments

x	character; strings to sort

Value

character

Examples

```
x <- c('pop_2020_est', 'pop_white_2020', 'pop_black_2020', 'pop_white_2021')
sort_suff(x)
```

Index