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 ................................................................. 2
add_suff ................................................................. 3
compare_names .......................................................... 3
list_phrase .............................................................. 4
### Description

Add Prefix

### Usage

`add_pref(x, pref)`

### Arguments

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

### Value

character

### Examples

```r
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

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

Description
List Prefixes

Usage
list_pref(tb)

Arguments
- tb: tibble; data to list prefixes in
**Value**

character

**Examples**

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

---

**Description**

List Suffixes

**Usage**

```r
list_suff(tb)
```

**Arguments**

- `tb` : tibble; data to list prefixes in

**Value**

character

**Examples**

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

---

**Description**

Relocate columns

**Usage**

```r
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

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

---

### rem_phrase

**Remove Phrase**

**Description**

Remove Phrase

**Usage**

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

**Arguments**

- `x` character; string to change
- `phrase` character; phrase to remove

**Value**

character

**Examples**

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

**Description**
Remove Prefix

**Usage**
remPref(x, pref)

**Arguments**
- x: character; string to change
- pref: character; prefix to remove

**Value**
character

**Examples**
```r
x <- c("pop", "pop_2020_est", "pop_white_2020", "pop_black_2020")
remPref(x, "pop_")
```

---

**remSuff**

**Description**
Remove Suffix

**Usage**
remSuff(x, suff)

**Arguments**
- x: character; string to change
- suff: character; suffix to remove

**Value**
character
rename_with_loud

Examples

```r
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**

```r
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**

```r
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**
```r
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(x, '_2020', '_20')

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


Examples

data(sd)

---

<table>
<thead>
<tr>
<th>sort_phrase</th>
<th>Sort by Phrase</th>
</tr>
</thead>
</table>

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

* compare
  compare_names, 3
  rename_with_loud, 8
* data
  sd, 10
* list
  list_phrase, 4
  list_pref, 4
  list_suff, 5
* phrase
  rem_phrase, 6
  repl_phrase, 9
* prefix
  add_pref, 2
  rem_pref, 7
  repl_pref, 9
* sort
  sort_phrase, 11
  sort_pref, 12
  sort_suff, 12
* suffix
  add_suff, 3
  rem_suff, 7
  repl_suff, 10

add_pref, 2
add_suff, 3

compare_names, 3

list_phrase, 4
list_pref, 4
list_suff, 5

relocate_with, 5
rem_phrase, 6
rem_pref, 7
rem_suff, 7
rename_with_loud, 8
repl_phrase, 9

repl_pref, 9
repl_suff, 10

sd, 10
sort_phrase, 11
sort_pref, 12
sort_suff, 12