

Package ‘tidier’

October 3, 2026

Title Enhanced 'mutate' with 'Apache Spark' Style Window Operations

Version 0.3.0

Description Window operations for R dataframes with 'by', 'order_by' and 'frame' defined by 'rows_between' or 'range_between', inspired by 'Apache Spark' via 'mutate' in 'dplyr' flavor.

Imports dplyr (>= 1.1.0), tidyr (>= 1.3.0), checkmate (>= 2.1.0), rlang (>= 1.0.6), slider (>= 0.2.2), magrittr (>= 1.5),

Suggests lubridate, stringr, testthat, duckdb, tibble, furr, DBI, future, dbplyr,

URL <https://github.com/talegari/tidier>

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 4.1.0)

NeedsCompilation no

Author Srikanth Komala Sheshachala [aut, cre]

Maintainer Srikanth Komala Sheshachala <sri.teach@gmail.com>

Repository CRAN

Date/Publication 2026-10-03 00:00:02 UTC

Contents

mutate	2
range_between	4
rows_between	5
Index	6

`mutate`*Drop-in replacement for [dplyr::mutate](#)*

Description

Provides supercharged version of [dplyr::mutate](#) with `.by` (group by), `.order_by` and `.frame` aggregation over arbitrary window frame

Usage

```
mutate(x, ..., .by, .order_by, .frame, .complete = FALSE)
```

Arguments

<code>x</code>	(<code>data.frame_</code>
<code>...</code>	expressions to be passed to dplyr::mutate
<code>.by</code>	(expression, optional: Yes) Columns to group by
<code>.order_by</code>	(expression, optional: Yes) Columns to order by
<code>.frame</code>	(vector, optional: Yes) Object of class <code>frame</code> created by one of these functions: <code>rows_between</code> , <code>range_between</code>
<code>.complete</code>	(flag, default: FALSE) passed to slider::slide

Details

A window function returns a value for every input row of a dataframe based on a group of rows (frame) in the neighborhood of the input row. This function implements computation over groups (partition_by in SQL) in a predefined order (order_by in SQL) across a neighborhood of rows (frame) defined by

- `rows_between`: Number of rows before and after the corresponding row. Example: `c(2, 1)`
- `range_between`: Range is numeric or interval objects Example: `c(days(2), days(1))`

This implementation is inspired by spark's [window API](#). The output has the same row order as the input independent of the `order_by`.

Value

`data.frame`

See Also

[rows_between\(\)](#), [range_between\(\)](#), [mutate\(\)](#)

Examples

```
library("magrittr") # for pipe
# example 1: rows between
# Using iris dataset,
# compute cumulative mean of column `Sepal.Length`
# ordered by `Petal.Width` and `Sepal.Width` columns
# grouped by `Petal.Length` column

iris %>%
  mutate(sl_mean = mean(Sepal.Length),
         .order_by = c(Petal.Width, Sepal.Width),
         .by = Petal.Length,
         .frame = rows_between(Inf, 0),
         ) %>%
  dplyr::slice_min(n = 3, Petal.Width, by = Species)

# example 2: range between
# Using a sample airquality dataset,
# compute mean temp over last seven days in the same month for every row

set.seed(101)
airquality %>%
  # create date column
  dplyr::mutate(date_col = lubridate::make_date(1973, Month, Day)) %>%
  # create gaps by removing some days
  dplyr::slice_sample(prop = 0.8) %>%
  # compute mean temperature over last seven days in the same month
  tidier::mutate(avg_temp_over_last_week = mean(Temp, na.rm = TRUE),
                 .order_by = date_col,
                 .by = Month,
                 .frame = range_between(
                   lubridate::days(7), # 7 days before current row
                   lubridate::days(-1) # do not include current row
                 )
                 )

# example 3: custom function / modeling over window frame
fit_lm_safe = function(df, formula, min_rows = 2) {
  if (is.null(df) || nrow(df) < min_rows) {
    return(NULL)
  }

  tryCatch(
    lm(formula, data = df),
    error = function(e) NULL
  )
}

mtcars %>%
  mutate(s = list(fit_lm_safe(pick(everything()), mpg ~ .)),
         .by = c(cyl, vs),
         .order_by = qsec,
```

```

      .frame = range_between(-1, 3),
      .complete = FALSE
    ) %>%
  head(10)

## Not run:
# example 4: parallel execution across many groups using tidyr::nest and furrr
future::plan(future::multisession(workers = 2))

iris %>%
  tidyr::nest(.by = Species) %>%
  dplyr::mutate(
    data = furrr::future_map(data, ~ .x %>%
      mutate(
        sl_mean = mean(Sepal.Length),
        .order_by = Petal.Width,
        .frame = rows_between(2, 2)
      )
    )
  ) %>%
  tidyr::unnest(data)

## End(Not run)

```

range_between

Create a frame indicating range between window

Description

Create a frame indicating range between window

Usage

```
range_between(before, after)
```

Arguments

before	range of rows before
after	range of rows after

Value

object of class range_between_frame, frame

See Also

[rows_between\(\)](#), [range_between\(\)](#), [mutate\(\)](#)

rows_between	Create a frame indicating rows between window
--------------	---

Description

Create a frame indicating rows between window

Usage

```
rows_between(before, after)
```

Arguments

before	Number of rows before
after	Number of rows after

Value

object of class rows_between_frame, frame

See Also

[rows_between\(\)](#), [range_between\(\)](#), [mutate\(\)](#)

Index

`dplyr::mutate`, [2](#)

`mutate`, [2](#)

`mutate()`, [2](#), [4](#), [5](#)

`range_between`, [4](#)

`range_between()`, [2](#), [4](#), [5](#)

`rows_between`, [5](#)

`rows_between()`, [2](#), [4](#), [5](#)

`slider::slide`, [2](#)