Package ‘neatRanges’

March 29, 2020

**Type** Package

**Title** Tidy Up Date/Time Ranges

**Version** 0.1.3

**Author** Aljaz Jelenko [aut, cre]

**Maintainer** Aljaz Jelenko <aljaz.jelenko@amis.net>

**BugReports** [https://github.com/arg0naut91/neatRanges/issues](https://github.com/arg0naut91/neatRanges/issues)

**Description** Collapse, partition, combine, fill gaps in and expand date/time ranges.

**URL** [https://github.com/arg0naut91/neatRanges](https://github.com/arg0naut91/neatRanges)

**License** MIT + file LICENSE

**Depends** R (>= 3.1.0)

**Imports** data.table, Rcpp (>= 1.0.3)

**LinkingTo** Rcpp

**Suggests** testthat

**Encoding** UTF-8

**RoxygenNote** 7.0.2

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2020-03-29 04:30:02 UTC

### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>collapse_ranges</td>
<td>2</td>
</tr>
<tr>
<td>combine_ranges</td>
<td>3</td>
</tr>
<tr>
<td>expand_dates</td>
<td>4</td>
</tr>
<tr>
<td>expand_times</td>
<td>5</td>
</tr>
<tr>
<td>fill_ranges</td>
<td>6</td>
</tr>
<tr>
<td>partition_ranges</td>
<td>7</td>
</tr>
</tbody>
</table>

**Index** 9
collapse_ranges

Collapses the consecutive date or timestamp ranges into one record.

Description
The date/time ranges where the gap between two records is equal to or less than max_gap parameter are collapsed into one record.

Usage

collapse_ranges(
  df,  
  groups = NULL,  
  start_var = NULL,  
  end_var = NULL,  
  startAttr = NULL,  
  endAttr = NULL,  
  dimension = "date",  
  max_gap = 0L,  
  fmt = "%Y-%m-%d",  
  tz = "UTC",  
  origin = "1970-01-01"
)

Arguments

df          Your data frame
groups      Grouping variables
start_var   Start of the range
end_var     End of the range
startAttr   Attributes linked to start of the range which should be kept (converted to character type by default)
endAttr     Attributes linked to end of the range which should be kept (converted to character type by default)
dimension   Indicate whether your range includes only dates ('date') or also timestamp ('timestamp'). Defaults to 'date'
max_gap     Gap between date or timestamp ranges, e.g. for 0, default, it will put together all records where there is no gap in-between
fmt         The format of your date or timestamp field, defaults to YMD
tz          Time zone, defaults to UTC
origin      Origin for timestamp conversion, defaults to 1970-01-01

Value
Returns a data frame (if initial input data.table, then data.table) with collapsed records.
combine_ranges

Examples

df_collapse <- data.frame(
id = c(rep("1111", 3), rep("2222", 3)),
rating = c("A+", "AA", "AA", rep("B-", 3)),
start_date = c("2014-01-01", "2015-01-01", "2016-01-01",
end_date = c("2014-12-31", "2015-12-31", "2016-03-01",
             "2017-01-31", "2018-12-31", "2020-02-01")
)
collapse_ranges(df_collapse, c("id", "rating"), "start_date", "end_date")

combine_ranges

Combines ranges from different tables into a single table.

Description

Combines ranges from different tables into a single table.

Usage

combine_ranges(
  dfs,
  groups = NULL,
  start_var = NULL,
  end_var = NULL,
  startAttr = NULL,
  endAttr = NULL,
  dimension = "date",
  max_gap = 0L,
  fmt = "%Y-%m-%d",
  tz = "UTC",
  origin = "1970-01-01"
)

Arguments

dfs A list of your data frames, e.g. list(df1, df2)
groups Grouping variables
start_var Start of the range
end_var End of the range
startAttr Attributes linked to start of the range which should be kept (converted to character type by default)
endAttr Attributes linked to end of the range which should be kept (converted to character type by default)
expand_dates

Description

Expand date ranges.

Usage

```r
expand_dates(
  df,
  start_var,
  end_var,
  name = "Expanded",
  fmt = "%Y-%m-%d",
  vars_to_keep = NULL,
  unit = "day"
)
```

Examples

```r
df1 <- data.frame(
  start = c("2010-01-01", "2012-06-01", "2014-10-15"),
  end = c("2010-08-05", "2013-03-03", "2015-01-01"),
  group = c("a", "a", "b"),
  infoScores = c(0, 3, 2)
)

df2 <- data.frame(
  end = c("2012-04-05", "2014-06-09", "2009-02-01"),
  group = c("b", "a", "b"),
  start = c("2009-01-15", "2012-07-08", "2008-01-01"),
  score = c(8, 2, 3)
)

combine_ranges(dfs = list(df1, df2), groups = "group",
               start_var = "start", end_var = "end")
```
**expand_times**

**Expand timestamp ranges.**

**Description**

Expand timestamp ranges.

**Usage**

```r
expand_times(
  df,
  start_var,
  end_var,
  name = "Expanded",
  fmt = "%Y-%m-%d %H:%M:%OS",
  vars_to_keep = NULL,
  unit = "hour",
  tz = "UTC"
)
```

**Arguments**

- `df` (Data frame (can also be a data.table or a tibble))
- `start_var` (Start Date column)
- `end_var` (End Date column)
- `name` (The name of newly created column. Defaults to `Expanded`)
- `fmt` (The format of date columns, defaults to Y-M-D)
- `vars_to_keep` (Which columns you would like to keep)
- `unit` (By which unit of time you want to expand; the default is day)

**Value**

Returns a full data frame with expanded sequences in a column, e.g. by day or month.

**Examples**

```r
df <- data.frame(
id = c("1111", "2222", "3333"),
gender = c("M", "F", "F"),
start = c("2018-01-01", "2019-01-01", "2020-01-01"),
end = c("2018-01-05", "2019-01-07", "2020-01-08")
)

expand_dates(df, start_var = "start", end_var = "end",
vars_to_keep = c("id", "gender"), unit = "day")
```

```r
df <- data.frame(
id = c("1111", "2222", "3333"),
gender = c("M", "F", "F"),
start = c("2018-01-01", "2019-01-01", "2020-01-01"),
end = c("2018-01-05", "2019-01-07", "2020-01-08")
)
expand_dates(df, start_var = "start", end_var = "end",
vars_to_keep = c("id", "gender"), unit = "day")
```
Arguments

- **df**: Data frame (can also be a data.table or a tibble)
- **start_var**: Start time column
- **end_var**: End time column
- **name**: The name of newly created column. Defaults to 'Expanded'
- **fmt**: The format of date columns, defaults to Y-M-D H:M:OS
- **vars_to_keep**: Which columns you would like to keep
- **unit**: By which unit of time you want to expand; the default is day
- **tz**: Desired time zone - defaults to UTC

Value

Returns a full data frame with expanded sequences in a column, e.g. by day or month.

Examples

```r
df <- data.frame(
id = c("1111", "2222", "3333"),
gender = c("M", "F", "F"),
start = c("2018-01-01 15:00:00", "2019-01-01 14:00:00", "2020-01-01 19:00:00"),
end = c("2018-01-01 18:30:00", "2019-01-01 17:30:00", "2020-01-02 02:00:00"))

expand_times(df, start_var = "start", end_var = "end",
vars_to_keep = c("id", "gender"), unit = "hour")
```

fill_ranges

*Fill the gaps between ranges.*

Description

Fill the gaps between ranges.

Usage

```r
fill_ranges(
  df,
  groups = NULL,
  start_var = NULL,
  end_var = NULL,
  fill = NULL,
  dimension = "date",
  fmt = "%Y-%m-%d",
  tz = "UTC",
  origin = "1970-01-01"
)
```
partition_ranges

Arguments

df                   Your data frame

groups               Grouping variables

start_var            Start of the range

end_var              End of the range

fill                 Fill the missing values for values corresponding to missing ranges, e.g. 'col-
                     name1 = 0, colname2 = Missing'

dimension            Indicate whether your range includes only dates ('date') or also timestamp ('times-
                     tamp'). Defaults to 'date'

fmt                  The format of your date or timestamp field, defaults to YMD

tz                   Time zone, defaults to UTC

origin               Origin for timestamp conversion, defaults to 1970-01-01

Value

Returns ordered data frame (if initial input data.table, then data.table) with added missing ranges.

Examples

df <- data.frame(
group = c("a", "a", "b", "b", "b"),
cost = c(143, 144, 105, 153, 124)
)

fill_ranges(df, start_var = "start", end_var = "end", groups = "group")

Description

Split ranges into multiple records

Usage

partition_ranges(
    df,
    start_var,
    end_var,
    fmt = "%Y-%m-%d",
    vars_to_keep = NULL,
    partition_by = "year"
)
Arguments

- **df**: Your data frame (can also be a data.table or a tibble)
- **start_var**: Start variable
- **end_var**: End variable
- **fmt**: Format of the date; defaults to Y-m-d
- **vars_to_keep**: Any column you’d like to retain (optional)
- **partition_by**: How should the range be partitioned (‘year’ or ‘month’); defaults to ‘year’

Value

Returns a data frame with start, end and optional grouping columns

Examples

```r
df <- data.frame(group = c("a", "a", "b", "b", "c"),
                end = c("2018-09-01", "2020-04-03", "2012-05-03", "2016-04-02", "2017-04-05")
)
partition_ranges(df, "start", "end", partition_by = "month")
```
Index

collapse_ranges, 2
combine_ranges, 3
expand_dates, 4
expand_times, 5
fill_ranges, 6
partition_ranges, 7