Package ‘groupr’
March 23, 2023

Title Groups with Inapplicable Values
Version 0.1.2
Description The ‘groupr’ package provides a more powerful version of grouped tibbles from ‘dplyr’. It allows groups to be marked inapplicable, which is a simple but widely useful way to express structure in a dataset. It also provides powerful pivoting and other group manipulation functions.
License MIT + file LICENSE
URL https://github.com/ngriffiths21/groupr
BugReports https://github.com/ngriffiths21/groupr/issues
Encoding UTF-8
Suggests testthat, knitr, rmarkdown, covr
RoxygenNote 7.2.3
Imports dplyr, rlang, vctrs, pillar, purrr, tidyr, tibble
VignetteBuilder knitr
NeedsCompilation no
Author Nicholas Griffiths [aut, cre] (<https://orcid.org/0000-0001-8166-9045>)
Maintainer Nicholas Griffiths <ngriffiths21@gmail.com>
Repository CRAN
Date/Publication 2023-03-23 03:50:02 UTC

R topics documented:
colgrp ................................................................. 2
group_by2 .......................................................... 2
infer_colgrps ...................................................... 3
pivot_grps ......................................................... 3
sep_colgrp ......................................................... 4
ungroup.igrouped_df ........................................... 5

Index 6
### colgrp

**Make a Single Column Grouping**

**Description**

Takes a tibble and groups columns together into a single data column. All columns that are not row indices will be grouped, and the resulting column will be named `data_name`.

**Usage**

```
colgrp(x, data_name, index_name = "group")
```

**Arguments**

- `x` A tibble
- `data_name` A string, the name of the new column
- `index_name` A string, the name of the new column index

**Value**

A grouped tibble

### group_by2

**Group a Tibble With Inapplicable Groups**

**Description**

Similar to `dplyr::group_by()`, this function groups a tibble while also marking certain groups as inapplicable.

**Usage**

```
group_by2(data, ...)
```

**Arguments**

- `data` A tibble to group
- `...` Arguments of the form `var = c(val1, val2)` or the name of a variable

**Details**

A grouped tibble has one or more grouping variables, where each unique combination of values identifies a group. This function allows some of the values to be marked inapplicable, such that the corresponding rows are not considered to be grouped on that variable at all.

Grouping variables, and inapplicable values, are passed as arguments in the form `group_var = c(value1, value2, ...)`. Any included values will be marked inapplicable. If an argument has length 0 or is NULL, no values will be marked inapplicable.
infer_colgrps

Value
An igrouped tibble

infer_colgrps
Set Column Grouping for a Structured Data Frame

Description
Takes a data frame where each non row index is a data frame column, and sets the corresponding column grouping.

Usage
infer_colgrps(x, index_name = "group", sep = "_")

Arguments
x A tibble with data frame data columns
index_name A name for the new column index
sep A character used to join the data column and group names

Value
A grouped tibble

pivot_grps
Pivot with Inapplicable Groups

Description
Pivot a dataset by defining the way the current grouping will be transformed into a new one. A pivot to wider consumes a row grouping (created by group_by2) and produces a new set of columns. A pivot to longer consumes a column grouping and produces a new row grouping.

Usage
pivot_grps(x, rows = NULL, cols = NULL)

Arguments
x A data frame
rows A list of character vectors, defining the new row grouping
cols A character vector, defining the new columns
Details

To pivot a column grouping to a row grouping, pass the specification of the new row grouping using the `cols` argument. The format is `list(values_col = "oldcol_1", "oldcol_2", ...)`. This will take all the data from the old columns, combine them into a new column `values_col`, and automatically provide a grouping variable, which will be called `name`. The values of `name` will be the corresponding names of the old columns.

To pivot a row grouping to a column grouping, pass a grouped dataset (using `group_by2`) and specify which grouping variable should be consumed to produce a set of new columns.

Both arguments can be passed in one call, in which case `rows` will be handled first, followed by `cols`.

See the introduction vignette for more details and examples.

Value

A pivoted data frame with the new grouping

---

```r
sep_colgrp(x, sep, index_name = "group")
```

Arguments

- `x` A tibble
- `sep` A character delimiting the two parts of the name
- `index_name` A name for the new column index

Value

A grouped tibble
Ungroup Method for Tibbles That Have Inapplicable Groups

Description

Ungroup method for tibbles that have inapplicable groups.

Usage

```r
# S3 method for class 'igrouped_df'
ungroup(x, ...
```

Arguments

- `x`: An igrouped tibble (as created by group_by)
- `...`: Ignored. All variables are removed from the grouping.

Value

A tibble with no groups. The "groups" attribute will be set to contain one column, .rows, with a single value that lists all rows.
Index

colgrp, 2

group_by2, 2

infer_colgrps, 3

pivot_grps, 3

sep_colgrp, 4

ungroup.igrouped_df, 5