

# Package ‘dedupewider’

September 20, 2021

**Title** Deduplication Across Multiple Columns

**Version** 0.0.2

**Description** Duplicated data can exist in different rows and columns and user may need to treat observations (rows) connected by duplicated data as one observation, e.g. companies can belong to one family (and thus: be one company) by sharing some telephone numbers. This package provides a function to find connected rows based on data on chosen columns and collapse it into one row.

**License** MIT + file LICENSE

**URL** <https://github.com/gsmolinski/dedupewider>

**BugReports** <https://github.com/gsmolinski/dedupewider/issues>

**Depends** R (>= 3.2.0)

**Imports** methods, data.table (>= 1.12.4)

**Suggests** rmarkdown, knitr, kableExtra, magrittr, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**VignetteBuilder** knitr

**Encoding** UTF-8

**RoxygenNote** 7.1.1

**NeedsCompilation** no

**Author** Grzegorz Smoliński [aut, cre]

**Maintainer** Grzegorz Smoliński <g.smolinski1@gmail.com>

**Repository** CRAN

**Date/Publication** 2021-09-20 20:40:02 UTC

## R topics documented:

dedupe_wide . . . . .	2
<b>Index</b>	<b>4</b>

---

dedupe_wide	<i>Dedupe across multiple columns</i>
-------------	---------------------------------------

---

### Description

Collapse many rows connected by duplicated data (which can exist in different rows and columns) into one, based on data in chosen columns, optionally putting non-consistent data into newly created additional columns.

### Usage

```
dedupe_wide(
  x,
  cols_dedupe,
  cols_expand = NULL,
  max_new_cols = NULL,
  enable_drop = TRUE
)
```

### Arguments

<code>x</code>	A data.frame without column named <code>'...idx'</code> and any column which ends by four dots and number (e.g. <code>'column....2'</code> ).
<code>cols_dedupe</code>	A character vector of length min. 2 of columns' names in <code>x</code> used to dedupe. Deduplicated data from these columns will be saved into new columns, number of which is control by <code>max_new_cols</code> .
<code>cols_expand</code>	A character vector of columns' names in <code>x</code> or <code>NULL</code> (means: none except those used to dedupe) indicating columns with data to keep in case of non-consistent data, i.e. unique data from these columns will be saved into new columns, number of which is control by <code>max_new_cols</code> .
<code>max_new_cols</code>	A numeric vector length 1 or <code>NULL</code> (means: without limit) indicating how many new columns can be created to store unique data from columns passed to <code>cols_dedupe</code> and each column passed to <code>cols_expand</code> . Cannot be lower than 1.
<code>enable_drop</code>	A logical vector length 1: should given column be dropped if (after deduplication) contains only missing data (NA)? Applicable only to columns used to dedupe.

### Details

Columns passed to `cols_dedupe` must be atomic.

Row names will always be removed. If you want to preserve row names, simply put in into separate column. Note that if this column won't be passed to `cols_expand` argument, only the one row name for duplicated rows will be preserved (row name closest to the top of the table).

Although [duplicated](#) or [unique](#) treats missing data (NA) as duplicated data, this function do not do this (see second example below).

Type of columns passed to `cols_dedupe` will be coerced to the most general type.

**Value**

If duplicated data found - data.frame with changed columns' names and optionally additional columns (in some cases less columns, depends on enable\_drop argument). Otherwise data.frame without changes (except row names removed).

**Note**

Internally, function is mainly based on `data.table` functions and thus enabling parallel computation is possible. To do this, just call `setDTthreads` before calling `dedupe_wide` function.

**Examples**

```
x <- data.frame(tel_1 = c(111, 222, 444, 555),
               tel_2 = c(222, 666, 666, 555),
               name = paste0("name", 1:4))
# rows 1, 2, 3 share the same phone numbers

dedupe_wide(x,
            cols_dedupe = c("tel_1", "tel_2"),
            cols_expand = "name")
# first three collapsed into one, for name4 kept only one phone number (555)
# 'name1', 'name2', 'name3' kept in new columns

y <- data.frame(tel_1 = c(777, 888, NA, NA),
               tel_2 = c(888, 777, NA, NA),
               name = paste0("name", 5:8))
# rows 3 and 4 has only missing data

dedupe_wide(y,
            cols_dedupe = c("tel_1", "tel_2"),
            cols_expand = "name")
# first two rows collapsed into one, nothing change for the rest of rows
```

# Index

`data.table`, [3](#)  
`dedupe_wide`, [2](#)  
`duplicated`, [2](#)  
  
`setDTthreads`, [3](#)  
  
`unique`, [2](#)