

# Package: safejoin (via r-universe)

September 10, 2024

**Title** Perform ``Safe" Table Joins

**Version** 0.1.0

**Description** The goal of 'safejoin' is to guarantee that when performing joins extra rows are not added to your data. 'safejoin' provides a wrapper around 'dplyr::left\_join' that will raise an error when extra rows are unexpectedly added to your data. This can be useful when working with data where you expect there to be a many to one relationship but you are not certain the relationship holds.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** false

**Suggests** testthat, knitr, rmarkdown

**Imports** dplyr, glue

**RoxygenNote** 7.1.1

**URL** <https://github.com/SamEdwardes/safejoin>

**BugReports** <https://github.com/SamEdwardes/safejoin/issues>

**VignetteBuilder** knitr

**Repository** <https://samedwardes.r-universe.dev>

**RemoteUrl** <https://github.com/samedwardes/safejoin>

**RemoteRef** HEAD

**RemoteSha** 2dced59275e9120a2798f2e88f7c2b95f063f756

## Contents

|                          |          |
|--------------------------|----------|
| safe_left_join . . . . . | 2        |
| <b>Index</b>             | <b>4</b> |

safe\_left\_join

*Validate extra rows are added not added to the left hand side***Description**

Perform a "safe" left join where it is guaranteed that no additional rows are added to the left hand side table. For more information on left joins see ([dplyr::left\\_join](#)).

**Usage**

```
safe_left_join(..., action = "error", relationship = "*:1")
```

**Arguments**

... Arguments passed on to [dplyr::left\\_join](#)

**x** A pair of data frames, data frame extensions (e.g. a tibble), or lazy data frames (e.g. from `dbplyr` or `dtplyr`). See *Methods*, below, for more details.

**y** A pair of data frames, data frame extensions (e.g. a tibble), or lazy data frames (e.g. from `dbplyr` or `dtplyr`). See *Methods*, below, for more details.

**by** A character vector of variables to join by.  
 If NULL, the default, `*_join()` will perform a natural join, using all variables in common across `x` and `y`. A message lists the variables so that you can check they're correct; suppress the message by supplying `by` explicitly. To join by different variables on `x` and `y`, use a named vector. For example, `by = c("a" = "b")` will match `x$a` to `y$b`.  
 To join by multiple variables, use a vector with length  $> 1$ . For example, `by = c("a", "b")` will match `x$a` to `y$a` and `x$b` to `y$b`. Use a named vector to match different variables in `x` and `y`. For example, `by = c("a" = "b", "c" = "d")` will match `x$a` to `y$b` and `x$c` to `y$d`.  
 To perform a cross-join, generating all combinations of `x` and `y`, use `by = character()`.

**copy** If `x` and `y` are not from the same data source, and `copy` is TRUE, then `y` will be copied into the same `src` as `x`. This allows you to join tables across `srcs`, but it is a potentially expensive operation so you must opt into it.

**suffix** If there are non-joined duplicate variables in `x` and `y`, these suffixes will be added to the output to disambiguate them. Should be a character vector of length 2.

**keep** Should the join keys from both `x` and `y` be preserved in the output?

**action** What should happen when the number of rows changes from a join? Options include: 'error', 'warning', or 'message'. By default 'error'.

**relationship** What is the expected relationship between 'x' and 'y'? At this time the only available option is '\*:1', indicating a many to one relationship between 'x' and 'y'. In the future more options may be added.

**Value**

An object of the same type as 'x'. The order of the rows and columns of 'x' is preserved as much as possible. The output has the following properties:

**Examples**

```
# The relationship between `x` and `y` is `*:1`. No extra rows will be added
# to the left hand side.
x <- data.frame(key = c("a", "a", "b"), value_x = c(1, 4, 2))
y <- data.frame(key = c("a", "b"), value_y = c(1, 1))
safe_left_join(x, y)

# The relationship between `x` and `y` is `1:*`. An error should be raised
# because additional rows will be added to the left hand side.
## Not run: x <- data.frame(key = c("a", "b"), value_x = c(1, 2))
y <- data.frame(key = c("a", "a"), value_y = c(1, 1))
safe_left_join(x, y)
## End(Not run)

# Alternatively instead of raising an error a warning or message can be
# outputted.
x <- data.frame(key = c("a", "b"), value_x = c(1, 2))
y <- data.frame(key = c("a", "a"), value_y = c(1, 1))
safe_left_join(x, y, action = "warning")
safe_left_join(x, y, action = "message")
```

# Index

`dplyr::left_join`, 2

`safe_left_join`, 2