

# Package ‘joyn’

April 28, 2021

**Type** Package

**Title** Tool for Diagnosis of Tables Joins and Complementary Join Features

**Version** 0.1.3

**Description** Tool for diagnosing table joins. It combines the speed of `data.table`, the flexibility of `dplyr`, and the diagnosis and features of the `merge` command in `Stata`.

**License** MIT + file LICENSE

**Encoding** UTF-8

**URL** <https://github.com/randrescastaneda/joyn>

**BugReports** <https://github.com/randrescastaneda/joyn/issues>

**Suggests** badger, covr, knitr, rmarkdown, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**Imports** rlang, data.table, glue, cli, stats, utils

**Depends** R (>= 2.10)

**RoxygenNote** 7.1.1

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** R.Andres Castaneda [aut, cre]

**Maintainer** R.Andres Castaneda <acastaneda@worldbank.org>

**Repository** CRAN

**Date/Publication** 2021-04-28 15:40:02 UTC

## R topics documented:

freq_table . . . . .	2
is_id . . . . .	3
merge . . . . .	3
possible_ids . . . . .	6

x1	7
x2	7
x3	7
x4	8
y1	8
y2	8
y3	9
y4	9

<b>Index</b>	<b>10</b>
--------------	-----------

---

freq_table	<i>tabulate simple frequencies</i>
------------	------------------------------------

---

### Description

tabulate one variable frequencies

### Usage

```
freq_table(x, byvar, digits = 1, na.rm = TRUE)
```

### Arguments

x	data frame
byvar	character: name of variable to tabulate. Use Standard evaluation.
digits	numeric: number of decimal places to display. Default is 1.
na.rm	logical: if TRUE remove NAs from calculations. Default is TRUE

### Value

data.table with frequencies.

### Examples

```
freq_table(x4, "id1")
```

---

is_id	<i>Make sure the match type is correct</i>
-------	--

---

**Description**

Make sure the match type is correct

**Usage**

```
is_id(dt, by, verbose = TRUE, return_report = FALSE)
```

**Arguments**

dt	either right of left table
by	by argument in merge
verbose	logical: if TRUE messages will be displayed
return_report	logical: if TRUE, returns data with summary of duplicates. If FALSE, returns logical value depending on whether dt is uniquely identified by by

**Value**

logical or data.frame, depending on the value of argument return\_report

**Examples**

```
is_id(y3, by = "id")
is_id(y3, by = "id", return_report = TRUE)
```

---

merge	<i>Merge two tables</i>
-------	-------------------------

---

**Description**

This is the main and, basically, the only function in joyn.

**Usage**

```
merge(
  x,
  y,
  by = intersect(names(x), names(y)),
  yvars = TRUE,
  match_type = c("m:m", "m:1", "1:m", "1:1"),
  keep = c("full", "left", "master", "right", "using", "inner"),
  update_values = FALSE,
```

```

update_NAs = update_values,
reportvar = "report",
reporttype = c("character", "numeric"),
roll = NULL,
keep_y_in_x = FALSE,
sort = TRUE,
verbose = getOption("joyn.verbose")
)

```

## Arguments

x	data frame: referred to <i>left</i> in R terminology, or <i>master</i> in Stata terminology.
y	data frame: referred to <i>right</i> in R terminology, or <i>using</i> in Stata terminology.
by	a character vector of variables to join by. If NULL, the default, joyn will do a natural join, using all variables with common names across the two tables. A message lists the variables so that you can check they're right (to suppress the message, simply explicitly list the variables that you want to join). To join by different variables on x and y use a vector of expressions. For example, by = c("a = b", "z") will use "a" in x, "b" in y, and "z" in both tables.
yvars	character: Vector of variable names that will be kept after the merge. If TRUE (the default), it keeps all the brings all the variables in y into x. If FALSE or NULL, it does not bring any variable into x, but a report will be generated.
match_type	character: one of "m:m", "m:1", "1:m", "1:1". Default is "m:m" since this is the default generally used in joins in R. However, following Stata's recommendation, it is better to be explicit and use any of the other three match types (See details in <i>match types sections</i> ).
keep	character: One of "full", "left", "master", "right", "using", "inner". Default is "full". Even though this is not the regular behavior of joins in R, the objective of joyn is to present a diagnosis of the join, so that it must use by default a full join. Yet, if "left" or "master", it keeps the observations that matched in both tables and the ones that did not match in x. The ones in y will be discarded. If "right" or "using", it keeps the observations that matched in both tables and the ones that did not match in y. The ones in x will be discarded. If "inner", it only keeps the observations that matched both tables.
update_values	logical: If TRUE, it will update all values of variables in x with the actual of variables in y with the same name as the ones in x. <b>NAs from y won't be used to update actual values in x.</b>
update_NAs	logical: If TRUE, it will update NA values of all variables in x with actual values of variables in y that have the same name as the ones in x. If FALSE, NA values won't be updated.
reportvar	character: Name of reporting variable. Default if "report". This is the same as variable "_merge" in Stata after performing a merge. If FALSE or NULL, the reporting variable will be excluded from the final table, though a summary of the join will be display after concluding.
reporttype	character: One of "character" or "numeric". Default is "character". If "numeric", the reporting variable will contain numeric codes of the source and the contents of each observation in the joined table.

roll	double: <i>to be implemented</i>
keep_y_in_x	logical: If TRUE, it will keep the original variable from y when both tables have common variable names. Thus, the prefix "y." will be added to the original name to distinguish from the resulting variable in the joined table.
sort	logical: If TRUE, sort by key variables in by. Default is TRUE.
verbose	logical: if FALSE, it won't display any message (programmer's option). Default is TRUE.

### Value

a data.table joining x and y.

### match types

Using the same wording of the [Stata manual](#)

**1:1:** specifies a one-to-one match merge. The variables specified in by uniquely identify single observations in both table.

**1:m and m:1:** specify *one-to-many* and *many-to-one* match merges, respectively. This means that in of the tables the observations are uniquely identify by the variables in by, while in the other table many (two or more) of the observations are identify by the variables in by

**m:m** refers to *many-to-many merge*. variables in by does not uniquely identify the observations in either table. Matching is performed by combining observations with equal values in by; within matching values, the first observation in the master (i.e. left or x) table is matched with the first matching observation in the using (i.e. right or y) table; the second, with the second; and so on. If there is an unequal number of observations within a group, then the last observation of the shorter group is used repeatedly to match with subsequent observations of the longer group.

### Examples

```
# Simple merge
library(data.table)
x1 = data.table(id = c(1L, 1L, 2L, 3L, NA_integer_),
               t = c(1L, 2L, 1L, 2L, NA_integer_),
               x = 11:15)

y1 = data.table(id = 1:2,
               y = c(11L, 15L))

x2 = data.table(id = c(1, 1, 2, 3, NA),
               t = c(1L, 2L, 1L, 2L, NA_integer_),
               x = c(16, 12, NA, NA, 15))

y2 = data.table(id = c(1, 2, 5, 6, 3),
               yd = c(1, 2, 5, 6, 3),
               y = c(11L, 15L, 20L, 13L, 10L),
               x = c(16:20))

merge(x1, y1)

# Bad merge for not specifying by argument
```

```
merge(x2, y2)

# good merge, ignoring variable x from y
merge(x2, y2, by = "id")

# update NAs in x variable form x
merge(x2, y2, by = "id", update_NAs = TRUE)

# Update values in x with variables from y
merge(x2, y2, by = "id", update_values = TRUE)
```

---

possible\_ids

*Find possible unique identifies of data frame*

---

### Description

Find possible unique identifies of data frame

### Usage

```
possible_ids(
  dt,
  exclude = NULL,
  include = NULL,
  verbose = getOption("possible_ids.verbose")
)
```

### Arguments

dt	data frame
exclude	character: Exclude variables to be selected as identifiers. It could be either the name of the variables of one type of the variable prefixed by "_". For instance, "_numeric" or "_character".
include	character: Name of variable to be included, that might belong to the group excluded in the exclude
verbose	logical: If FALSE no message will be displayed. Default is TRUE

### Value

list with possible identifiers

### Examples

```
possible_ids(x4)
```

---

x1	<i>Testing data x1</i>
----	------------------------

---

**Description**

testing data

**Usage**

x1

**Format**

A data frame with 5 rows and 3 variables:

---

x2	<i>Testing data x2</i>
----	------------------------

---

**Description**

testing data

**Usage**

x2

**Format**

A data frame with 5 rows and 3 variables:

---

x3	<i>Testing data x3</i>
----	------------------------

---

**Description**

testing data

**Usage**

x3

**Format**

A data frame with 3 rows and 3 variables:

---

x4                      *Testing data x4*

---

**Description**

testing data

**Usage**

x4

**Format**

A data frame with 5 rows and 4 variables:

---

y1                      *Testing data y1*

---

**Description**

testing data

**Usage**

y1

**Format**

A data frame with 3 rows and 2 variables:

---

y2                      *Testing data y2*

---

**Description**

testing data

**Usage**

y2

**Format**

A data frame with 5 rows and 4 variables:



---

y3	<i>Testing data y3</i>
----	------------------------

---

**Description**

testing data

**Usage**

y3

**Format**

A data frame with 4 rows and 2 variables:

---

y4	<i>Testing data y4</i>
----	------------------------

---

**Description**

testing data

**Usage**

y4

**Format**

A data frame with 5 rows and 3 variables:

# Index

## \* datasets

x1, 7

x2, 7

x3, 7

x4, 8

y1, 8

y2, 8

y3, 9

y4, 9

freq\_table, 2

is\_id, 3

merge, 3

possible\_ids, 6

x1, 7

x2, 7

x3, 7

x4, 8

y1, 8

y2, 8

y3, 9

y4, 9