Package ‘diffdf’

September 24, 2024

Type Package

Title Dataframe Difference Tool

Version 1.1.1

Description Functions for comparing two data.frames against each other. The core functionality is to provide a detailed breakdown of any differences between two data.frames as well as providing utility functions to help narrow down the source of problems and differences.

Encoding UTF-8

Language en-GB

Depends R (>= 3.1.2)

Imports tibble, assertthat, methods

Suggests testthat, lubridate, knitr, rmarkdown, purrr, dplyr, stringi, stringr, devtools, covr, bit64

RoxygenNote 7.3.2

VignetteBuilder knitr

License MIT + file LICENSE

URL https://gowerc.github.io/diffdf/,
https://github.com/gowerc/diffdf/

Config/testthat/edition 3

BugReports https://github.com/gowerc/diffdf/issues

NeedsCompilation no

Author Craig Gower-Page [cre, aut],
Kieran Martin [aut]

Maintainer Craig Gower-Page <craig.gower-page@roche.com>

Repository CRAN

Date/Publication 2024-09-24 17:00:02 UTC

Contents

`as_character` *as_character*

Description

Stub function to enable mocking in unit tests

Usage

```
as_character()
```

`diffdf` *diffdf*

Description

Compares 2 dataframes and outputs any differences.

Usage

```
diffdf(
  base,
  compare,
  keys = NULL,
  suppress_warnings = FALSE,
  strict_numeric = TRUE,
  strict_factor = TRUE,
  file = NULL,
  tolerance = sqrt(.Machine$double.eps),
  scale = NULL,
  check_column_order = FALSE,
  check_df_class = FALSE
)
```

Arguments

`base` input dataframe

`compare` comparison dataframe

`keys` vector of variables (as strings) that defines a unique row in the base and compare dataframes

`suppress_warnings` Do you want to suppress warnings? (logical)

`strict_numeric` Flag for strict numeric to numeric comparisons (default = TRUE). If False diffdf will cast integer to double where required for comparisons. Note that variables specified in the keys will never be casted.

`strict_factor` Flag for strict factor to character comparisons (default = TRUE). If False diffdf will cast factors to characters where required for comparisons. Note that variables specified in the keys will never be casted.

`file` Location and name of a text file to output the results to. Setting to NULL will cause no file to be produced.

`tolerance` Set tolerance for numeric comparisons. Note that comparisons fail if (x-y)/scale > tolerance.

`scale` Set scale for numeric comparisons. Note that comparisons fail if (x-y)/scale > tolerance. Setting as NULL is a slightly more efficient version of scale = 1.

`check_column_order` Should the column ordering be checked? (logical)

`check_df_class` Do you want to check for differences in the class between `base` and `compare`? (logical)

Examples

```
x <- subset(iris, -Species)
x[1, 2] <- 5
COMPARE <- diffdf(iris, x)
print(COMPARE)

#### Sample data frames

DF1 <- data.frame(
    id = c(1, 2, 3, 4, 5, 6),
    v1 = letters[1:6],
    v2 = c(NA, NA, 1, 2, 3, NA)
)

DF2 <- data.frame(
    id = c(1, 2, 3, 4, 5, 7),
    v1 = letters[1:6],
    v2 = c(NA, NA, 1, 2, NA, NA),
    v3 = c(NA, NA, 1, 2, NA, 4)
)

diffdf(DF1, DF1, keys = "id")
```

```
# We can control matching with scale/location for example:

DF1 <- data.frame(
    id = c(1, 2, 3, 4, 5, 6),
    v1 = letters[1:6],
    v2 = c(1, 2, 3, 4, 5, 6)
)
DF2 <- data.frame(
    id = c(1, 2, 3, 4, 5, 6),
    v1 = letters[1:6],
    v2 = c(1.1, 2, 3, 4, 5, 6)
)

diffdf(DF1, DF2, keys = "id")
diffdf(DF1, DF2, keys = "id", tolerance = 0.2)
diffdf(DF1, DF2, keys = "id", scale = 10, tolerance = 0.2)

# We can use strict_factor to compare factors with characters for example:

DF1 <- data.frame(
    id = c(1, 2, 3, 4, 5, 6),
    v1 = letters[1:6],
    v2 = c(NA, NA, 1, 2, 3, NA),
    stringsAsFactors = FALSE
)

DF2 <- data.frame(
    id = c(1, 2, 3, 4, 5, 6),
    v1 = letters[1:6],
    v2 = c(NA, NA, 1, 2, 3, NA)
)

diffdf(DF1, DF2, keys = "id", strict_factor = TRUE)
diffdf(DF1, DF2, keys = "id", strict_factor = FALSE)
```

`diffdf_has_issues` *diffdf_has_issues*

Description

Utility function which returns TRUE if an diffdf object has issues or FALSE if an diffdf object does not have issues

Usage

```
diffdf_has_issues(x)
```

Arguments

x	diffdf object

Examples

```
# Example with no issues
x <- diffdf(iris, iris)
diffdf_has_issues(x)

# Example with issues
iris2 <- iris
iris2[2, 2] <- NA
x <- diffdf(iris, iris2, suppress_warnings = TRUE)
diffdf_has_issues(x)
```

diffdf_issuerows *Identify Issue Rows*

Description

This function takes a `diffdf` object and a dataframe and subsets the `data.frame` for problem rows as identified in the comparison object. If `vars` has been specified only issue rows associated with those variable(s) will be returned.

Usage

```
diffdf_issuerows(df, diff, vars = NULL)
```

Arguments

df	dataframe to be subsetted
diff	diffdf object
vars	(optional) character vector containing names of issue variables to subset dataframe on. A value of NULL (default) will be taken to mean available issue variables.

Details

Note that `diffdf_issuerows` can be used to subset against any dataframe. The only requirement is that the original variables specified in the keys argument to diffdf are present on the dataframe you are subsetting against. However please note that if no keys were specified in diffdf then the row number is used. This means using `diffdf_issuerows` without a keys against an arbitrary dataset can easily result in nonsense rows being returned. It is always recommended to supply keys to diffdf.

Examples

```
iris2 <- iris
for (i in 1:3) iris2[i, i] <- 99
x <- diffdf(iris, iris2, suppress_warnings = TRUE)
diffdf_issuerows(iris, x)
diffdf_issuerows(iris2, x)
diffdf_issuerows(iris2, x, vars = ”Sepal.Length”)
diffdf_issuerows(iris2, x, vars = c(”Sepal.Length”, ”Sepal.Width”))
```

`print.diffdf` *Print diffdf objects*

Description

Print nicely formatted version of an diffdf object

Usage

```
## S3 method for class 'diffdf'
print(x, row_limit = 10, as_string = FALSE, ...)
```

Arguments

`x`	comparison object created by diffdf().
`row_limit`	Max row limit for difference tables (NULL to show all rows)
`as_string`	Return printed message as an R character vector?
`...`	Additional arguments (not used)

Examples

```
x <- subset(iris, -Species)
x[1, 2] <- 5
COMPARE <- diffdf(iris, x)
print(COMPARE)
print(COMPARE, row_limit = 5)
```

Index