Package 'taber'

October 14, 2022

Type Package

Title Split and Recombine Your Data

Version 0.1.2

Date 2020-09-21

Maintainer Seth Wenchel <seth@wenchel.com>

Description Sometimes you need to split your data and work on the two chunks independently before bringing them back together. 'Taber' allows you to do that with its two functions.

License BSD_3_clause + file LICENSE

LazyData TRUE

Imports magrittr, dplyr

URL https://github.com/restonslacker/taber

BugReports https://github.com/restonslacker/taber/issues

RoxygenNote 7.1.1

NeedsCompilation no

Author Seth Wenchel [aut, cre, cph] (<https://orcid.org/0000-0001-6099-0200>)

Repository CRAN

Date/Publication 2020-10-04 11:40:02 UTC

R topics documented:

---

clear_stack *Remove all objects from the stack by deleting them from memory.*

Description

Remove all objects from the stack by deleting them from memory.

Usage

```
clear_stack()
```

Examples

```
library(dplyr)
aframe <- data.frame(zed = runif(100))
set_to_zero <- . %>% mutate(zed = 0)
aframe %>% scion(zed >0.5, false_fun=set_to_zero)
clear_stack()
```

---

graft *Graft*

Description

Graft one dataset onto another

Usage

```
graft(.data, combine_fun, data2)
```

Arguments

.data	A tbl or something that can be coerced into one
combine_fun	optional, A function that will combine two tbls such as full_join or bind_rows
data2	A tbl or something that can be coerced into one

Details

Graft requires two data objects. The first must be provided by the user. The second can either be passed in or automatically pulled off of the package's internal stack of scions. These will be combined accoring to the following rules in order:

- If either dataset has zero rows, the other dataset will be returned.
- If combine_fun is specifed, `combine_fun(.data, data2)` will be called
- If all column names match, a row bind will occur
- If at least some column names match, a full join will occur
- If both have the same number of rows a column bind will be performed

Value

A single tbl object

Author(s)

Seth Wenchel

Examples

```
library(dplyr)
aframe <- data.frame(zed = runif(100))
set_to_zero <- . %>% mutate(zed = 0)
aframe %>% scion(zed >0.5, false_fun=set_to_zero) %>% mutate(zed=1) %>% graft
```

---

scion	*scion*

Description

scion

Usage

```
scion(.data, ..., false_fun, false_name, false_env)
```

Arguments

`.data`	A tbl or something that can be coerced into one
`...`	conditions that will be passed to dplyr::filter
`false_fun`	A function or functional that will be applied to the data that doesn't pass the supplied filters (the scion)
`false_name`	optional, the name of the object to which the scion will be assigned.
`false_env`	optional, the environment into which the scion will be assigned. If specified, false_name must also be specified. If unspecified (default), scions will be placed into the internal package environment.

Details

`.data` will be split into two chunks based on the conditions. The scion will be passed through `false_fun` and then either placed on the package's internal stack or assigned as specified by `false_name` and `false_env`.

Value

A tbl whose rows have passed the stated conditions

Author(s)

Seth Wenchel

Examples

```
library(dplyr)
aframe <- data.frame(zed = runif(100))
set_to_zero <- . %>% mutate(zed = 0)
aframe %>% scion(zed >0.5, false_fun=set_to_zero) %>% mutate(zed=1) %>% graft
```

---

stack_view   *See what's on the stack*

Description

This is primarily to help with debugging.

Usage

```
stack_view(x)
```

Arguments

x   optional string. If supplied it should match the name of an object in the pack-
                 age enviroment. The value of the corresponding variable will be returned. If
                 missing, a list of all objects in the package enviroment.

Note

Note that [graft](#) does not delete objects from the environment.

Index