Package 'joinEasy'

April 30, 2022

Type Package

Title Performs Join Operations

Version 1.0

Date 2022-05-02

Author Blake Rosenberg

Maintainer Blake Rosenberg <Github@OpenAnIssue.com>

Description

Provides succinct solutions for simplifying the process of various join operations in R.

Depends sqldf

License MIT

R topics documented:

`anti_join` *Anti Join*

Description

This function returns the set of all tuples that are in neither the intersection of the two input relations nor in the right dataframe.

Usage

```
{
    # This function can be used without providing a primary key value:
    anti_join(left, right)
    # Or with a primary key value:
    anti_join(left, right, key)
}
```

Arguments

left	A dataframe which will be used as the left-side table in the operation.
right	A dataframe which will be used as the right-side table in the operation.
key	The primary key value. If not provided, the function will use the dataframes' row numbers as the primary key.

Value

rtn	The resulting dataframe.

Examples

```
# Example usage:
left <- data.frame(id = 100:106, letter = c('a', 'b', 'c', 'd', 'e', 'f', 'g'))
random_i <- sample(seq_len(nrow(left)))
cutpoint <- round(nrow(left) / 2)
right <- left[random_i[seq_len(cutpoint)],]
output <- anti_join(left, right, key = "id")
output
# right for comparison:
right
```

---

cross_join *Cross Join*

Description

This function returns the cartesian product of two relations.

Usage

```
cross_join(left, right)
```

Arguments

left	A dataframe which will be used as the left-side table in the operation.
right	A dataframe which will be used as the right-side table in the operation.

Value

rtn	The resulting dataframe.

Examples

```
# Example usage:
left <- data.frame(id = 100:106, letter = c('a', 'b', 'c', 'd', 'e', 'f', 'g'))
random_i <- sample(seq_len(nrow(left)))
cutpoint <- round(nrow(left) / 2)
right <- left[random_i[seq_len(cutpoint)],]
output <- cross_join(left, right)
output
```

`full_outer_exclusive_join` *Full Outer (INNER Exclusive) Join*

Description

The 'full_outer_exclusive_join' function is not yet implemented. Once released, this function will return all rows from both left and right which are unique to the dataframes.

Usage

```
{
    # This function will be usable without providing a primary key value:
    full_outer_exclusive_join(left, right)
    # Or with a primary key value:
    full_outer_exclusive_join(left, right, key)
}
```

Arguments

`left`	A dataframe which will be used as the left-side table in the operation.
`right`	A dataframe which will be used as the right-side table in the operation.
`key`	The primary key value. If not provided, the function will use the dataframes' row numbers as the primary key.

Value

`rtn`	The resulting dataframe.

Examples

```
# Planned example usage:
left <- data.frame(id = 100:106, letter = c('a', 'b', 'c', 'd', 'e', 'f', 'g'))
random_i <- sample(seq_len(nrow(left)))
cutpoint <- round(nrow(left) / 2)
right <- left[random_i[seq_len(cutpoint)],]
output <- full_outer_exclusive_join(left, right, key = "id")
output
```

`inner_join` *Inner Join*

Description

This function returns only those rows which are common to both dataframes.

Usage

```
{
    # This function can be used without providing a primary key value:
    inner_join(left, right)
    # Or with a primary key value:
    inner_join(left, right, key)
}
```

Arguments

left	A dataframe which will be used as the left-side table in the operation.
right	A dataframe which will be used as the right-side table in the operation.
key	The primary key value. If not provided, the function will use the dataframes' row numbers as the primary key.

Value

rtn	The resulting dataframe.

Examples

```
# Example usage:
left <- data.frame(id = 100:106, letter = c('a', 'b', 'c', 'd', 'e', 'f', 'g'))
random_i <- sample(seq_len(nrow(left)))
cutpoint <- round(nrow(left) / 2)
right <- left[random_i[seq_len(cutpoint)],]
output <- inner_join(left, right, key = "id")
output
```

---

joinEasy *Performs Join Operations*

Description

Provides succinct solutions for simplifying the process of various join operations in R.

Details

The DESCRIPTION file: This package was not yet installed at build time.

Index: This package was not yet installed at build time.
anti_join cross_join

Author(s)

Blake Rosenberg

Maintainer: Blake Rosenberg ¡Github@OpenAnIssue.com¿

References

www.mtitek.com/tutorials/oracle/sql-join.php Provides visual depictions of all operations present in this package

See Also

#˜˜ Optional links to other man pages, e.g. ˜˜ #˜˜ `<pkg>` ˜˜

Examples

```
# simple examples of the most important functions
```

`left_outer_join` *Left Outer Join*

Description

This function returns only those rows which are either common to both dataframes or only present in the left dataframe.

Usage

```
{
    # This function can be used without providing a primary key value:
    left_outer_join(left, right)
    # Or with a primary key value:
    left_outer_join(left, right, key)
}
```

Arguments

`left`	A dataframe which will be used as the left-side table in the operation.
`right`	A dataframe which will be used as the right-side table in the operation.
`key`	The primary key value. If not provided, the function will use the dataframes' row numbers as the primary key.

Value

`rtn`	The resulting dataframe.

Examples

```
# Example usage:
left <- data.frame(id = 100:106, letter = c('a', 'b', 'c', 'd', 'e', 'f', 'g'))
random_i <- sample(seq_len(nrow(left)))
cutpoint <- round(nrow(left) / 2)
right <- left[random_i[seq_len(cutpoint)],]
output <- left_outer_join(left, right, key = "id")
output
```

`right_outer_join` *Right Outer Join*

Description

The 'right_outer_join' function is not yet implemented. Once released, this function will return only those rows which are either common to both dataframes or only present in the right dataframe.

Usage

```
{
    # This function will be usable without providing a primary key value:
    right_outer_join(left, right)
    # Or with a primary key value:
    right_outer_join(left, right, key)
}
```

Arguments

`left`	A dataframe which will be used as the left-side table in the operation.
`right`	A dataframe which will be used as the right-side table in the operation.
`key`	The primary key value. If not provided, the function will use the dataframes' row numbers as the primary key.

Value

`rtn`	The resulting dataframe.

Examples

```
# Planned example usage:
left <- data.frame(id = 100:106, letter = c('a', 'b', 'c', 'd', 'e', 'f', 'g'))
random_i <- sample(seq_len(nrow(left)))
cutpoint <- round(nrow(left) / 2)
right <- left[random_i[seq_len(cutpoint)],]
output <- right_outer_join(left, right, key = "id")
output
```

`semi_join` *Semi Join*

Description

This function returns a similar result to that of the inner_join, except that this one never duplicates rows.

Usage

```
{
    # This function can be used without providing a primary key value:
    semi_join(left, right)
    # Or with a primary key value:
    semi_join(left, right, key)
}
```

Arguments

`left`	A dataframe which will be used as the left-side table in the operation.
`right`	A dataframe which will be used as the right-side table in the operation.
`key`	The primary key value. If not provided, the function will use the dataframes' row numbers as the primary key.

Value

`rtn`	The resulting dataframe.

Examples

```
# Example usage:
left <- data.frame(id = 100:106, letter = c('a', 'b', 'c', 'd', 'e', 'f', 'g'))
random_i <- sample(seq_len(nrow(left)))
cutpoint <- round(nrow(left) / 2)
right <- left[random_i[seq_len(cutpoint)],]
output <- semi_join(left, right, key = "id")
output
```