'tidbitR'

April 23, 2017

`ClipbrdRead` *Copies from clipboard*

Description

Copies a tsv table from clipboard.

Usage

```
ClipbrdRead(sep = "\t", quote = "\"", stringsAsFactors = F, ...)
```

Arguments

`sep`	Column separator. Defaulted to \t.

Value

Gets clipboard.

Examples

```
ClipbrdRead()
ds <- ClipbrdRead(quote = "'")
```

`ClipbrdWrite` *Copies to clipboard*

Description

Copies an object to clipboard as a tsv table.

Usage

```
ClipbrdWrite(x, sep = "\t", quote = T)
```

Arguments

x	tbl to copy.
sep	Column separator. Defaulted to \t.

Value

Copies the object to the clipboard.

Examples

```
ClipbrdWrite(ds)
ds %>%
  ClipbrdWrite()
```

Compare *Compare two dataframe for differences*

Description

Compare two data frames. Both the data frames must have the same set of columns.

Usage

```
Compare(x, y, inXbutNotY = T)
```

Arguments

x	First data frame to compare.
y	Second data frame to compare with.
inXbutNotY	If True (default) informs to find differences in X and not in Y. If False infors to find differences in Y and not in X.

Value

Returns a tibble with the difference.

Examples

```
Compare(x, y)
Compare(y, x)
```

CompareAndShowAll *Compare two dataframe to show differences highlighted*

Description

Compares two data frames to show two additional columns x_match, y_match to inform the number of rows that matches against x and y. These columns informs if the row is present in x or y or both. Both the data frames must have the same set of columns.

Usage

```
CompareAndShowAll(x, y, col.names = c("lhs_matches", "rhs_matches"))
```

Arguments

x	First data frame to compare.
y	Second data frame to compare with.

Value

Returns a tibble with the difference.

CompareOraDataset *Compares two SQL resultsets from two database environments*

Description

Takes a SQL statement as an input and executes the SQL in two database environments, then compares the SQL resultset for differences.

Usage

```
CompareOraDataset(query, env1, env2, inXbutNotY = T)
```

Arguments

query	SQL Query.
env1	Environment string to the first database.
env2	Environment string to the second database.
inXbutNotY	If True (default) informs to find differences in X and not in Y. If False infors to find differences in Y and not in X.

Value

Returns a tibble of differences

Crunch *Shows a quick summary of a data frame*

Description

Shows element type, row count, na row count, unique row count, mean, min, Q1, median, Q3, max, min string length, max string length and set of samples from each of the columns in a data frame.

Usage

```
Crunch(df, sample = 20, sample_delim = "; ")
```

Arguments

df	Data frame.
sample	Number of samples from each column. Defaults to 20.
sample_delim	Delimiter for each sample.

ODBCRun *Runs a SQL query using ODBC driver*

Description

Runs a SQL query in a database using the ODBC driver and returns a resultset. Before hand a DSN has to be created under Control Panel -> Administrative tools.

Usage

```
ODBCRun(query, dsn = "mydsn")
```

Arguments

query	SQL Query to execute.
dsn	Data source name.

Value

SQL resultset in a dataframe

Examples

```
ODBCRun("SELECT * FROM ALL_TABLES", "myDSN")
```

OraRun *Runs a SQL query in preferred environment.*

Description

Runs a SQL query in Oracle and returns a resultset.

Usage

```
OraRun(query, env = "dev")
```

Arguments

query	SQL Query to execute.
env	Environment name as string. The function will parse the environment name to its equivalent connection string stored under ora.connstr.[env]. This connection must be defined in advance that must contain five variables host_name, port, sid, user_name, pwd. Defaults to environment dev. E.g., ora.connstr.dev <- list(host_name = "localhost", port = "1521", sid = "xe", user_name = "scott", pwd = "tiger")

Value

SQL resultset as a tibble

Examples

```
ora.connstr.stg <- list(host_name = "localhost", port = "1521", sid = "xe", user_name = "scott", pwd = "tiger"
OraRun("SELECT * FROM ALL_TABLES", "stg")
```

OraRun_ *Runs a SQL query in Oracle*

Description

Runs a SQL query in Oracle and returns a resultset.

Usage

```
OraRun_(query, host_name, port = "1521", sid = "xe", user_name, pwd)
```

Arguments

query	SQL Query to execute.
host_name	Host name.
port	Port name. Defaults to port 1521.
sid	Service Id. Defaults to xe.
user_name	User name for the connection in plain text.
pwd	Password for the connection in plain text.

Value

SQL resultset in a tibble

Examples

```
OraRun_("SELECT * FROM ALL_TABLES", "act", 1580, xe, "scott", "tiger")
```

`OraTableDesc` *Gets table description from Oracle db*

Description

Table description of an Oracle table is retrieved in a concise format.

Usage

```
OraTableDesc(db_table, env = "dev")
```

Arguments

`db_table`	Name of the database table.
`env`	Environment string to use for forming the connection string.

Value

Returns a data frame

Examples

```
OraTableDesc("EMP", "dev")
```

`StringToDate` *Converts an date string to a date*

Description

Converts a string in date format(dd-MMM-yy) into a R Date.

Usage

```
StringToDate(x, century = F)
```

Arguments

`x`	Date string to be converted.
`century`	Informs if the string has the century included (dd-MM-yyyy). Default is F.

Value

Returns the date string as a date

Examples

```
StringToDate("01-JAN-16")
StringToDate("01-JAN-2017", T)
```

`XmlGetValues` *Gets all XML node text*

Description

Parses through all XML end nodes and returns the text of those nodes in a tibble.

Usage

```
XmlGetValues(x, nodepath = "*")
```

Arguments

`x`	XML document.
`nodepath`	Xpath to node for which text has to be collected. Defaults to * that will collect text of all nodes.

Value

Returns xpath to a node, node name and text value in a tibble.

Examples

```
XmlGetValues(xml2::read_xml("https://www.w3schools.com/xml/simple.xml")) %>%
  View()
XmlGetValues(xml2::read_xml("https://www.w3schools.com/xml/simple.xml"), "calories") %>%
  View()
```

Index