Package ‘sqlutils’

February 20, 2015

Type Package

Title Utilities for working with SQL files.

Version 1.2

Date 2014-11-19

Author Jason Bryer

Maintainer Jason Bryer <jason@bryer.org>

URL http://jason.bryer.org/sqlutils, http://github.com/jbryer/sqlutils

BugReports https://github.com/jbryer/sqlutils/issues

Description This package provides utilities for working with a library of SQL files.

License GPL

Depends roxygen2, stringr, DBI

Suggests tcltk, sqldf, xtable

Enhances RPostgreSQL, RODBC, RMySQL, RSQLite, RJDBC

NeedsCompilation no

Repository CRAN

Date/Publication 2014-11-19 17:49:01

R topics documented:

`sqlutils-package` *Utilities for working with SQL files.*

Description

Utilities for managing a library of SLQ files.

Author(s)

Jason Bryer <`jason@bryer.org`>

`cacheQuery` *Function for working with cached queries.*

Description

This will first look in the given directory for a CSV or Rda version of the file, if it exists, that will be read and returned. Otherwise it will execute the query and then saves a CSV or Rda file.

Usage

```
cacheQuery(query = NULL, dir = getwd(), filename = getCacheFilename(query
  = query, dir = dir, ext = format, ...), format = "rda", maxLevels = 20,
  ...)
```

Arguments

`dir`	the directory to save and load cached data files. Defaults to the current working directory (i.e. `getwd`.
`filename`	the filename of the cached data file.
`query`	the query to execute.
`maxLevels`	the maximum number of levels a factor can have before being converted to a character vector.
`...`	other parameters passed to the `execQuery` function including query parameters.
`format`	either `csv` for comma separated value files or `rda` for R data files.

Value

a data frame.

`execQuery` *Executes the specified query and returns a data frame. This function currently supports RODBC, RSQLite, and RMySQL. For other databases, use getQuery() and execute the SQL statement using the appropriate database connection.*

Description

Executes the specified query and returns a data frame. This function currently supports RODBC, RSQLite, and RMySQL. For other databases, use getQuery() and execute the SQL statement using the appropriate database connection.

Usage

```
execQuery(query = NULL, connection = NULL, maxLevels = 20, ...)
```

Arguments

`query`	the query to execute.
`connection`	the database connection.
`maxLevels`	the maximum number of levels a factor can have before being converted to a character. Set to `NULL` to not recode.
`...`	other parameters passed to `getSQL` and `sqlexec`.

See Also

sqlexec, cacheQuery

getCacheFilename *Returns the complete filepath to the cache file.*

Description

Returns the complete filepath to the cache file.

Usage

```
getCacheFilename(query, dir = getwd(), ext = "csv", ...)
```

Arguments

`query`	the query name.
`dir`	the directory to save the cache file to.
`ext`	file extension.
`...`	query parameters.

Value

full filepath to the cached file.

getParameters *Returns the parameters that must be set for the given query.*

Description

Returns the parameters that must be set for the given query.

Usage

```
getParameters(query)
```

Arguments

`query`	the query name.

Value

list of parameter names.

getQueries *Returns a list of available queries in the current repository.*

Description

Returns a list of available queries in the current repository.

Usage

```
getQueries()
```

getSQL *Returns the query as a string with the parameters set.*

Description

Returns the query as a string with the parameters set.

Usage

```
getSQL(query = NULL, ...)
```

Arguments

query	the query name.
...	SQL parameters.

Value

the SQL string with parameters replaced.

getSQLRaw *Returns the SQL from the file without the parameters replaced.*

Description

Returns the SQL from the file without the parameters replaced.

Usage

```
getSQLRaw(query)
```

Arguments

`query`	the query name.

Value

the unedited SQL statement.

`is.null.string` *Does the string contain no matter, but very well [:space:]?*

Description

Does the string contain no matter, but very well [:space:]?

Usage

```
is.null.string(string)
```

Arguments

`string`	the string to check

Value

TRUE if the string contains words, otherwise FALSE

`isql` *Interactive SQL session.*

Description

This function will start an interactive SQL session. The user can enter SQL statements and execute them against the given database connection. This was initially developed as a teaching tool for learning SQL.

Usage

```
isql(conn, sql = character(), envir = baseenv(), ...)
```

Arguments

`conn`	a database connection.
`sql`	initial SQL statement.
`envir`	the environment to save data frames when executing `save`.
`...`	other parameters passed to `sqlexec`.

Value

returns a list containing two character vectors, one with a history of commands and another with a history of SQL statements.

`parse.element` *Parse a raw string containing key and expressions.*

Description

Copied from roxygen2: https://github.com/yihui/roxygen2/blob/master/R/parse-preref.R

Usage

```
parse.element(element, srcref)
```

Arguments

`element`	the string containing key and expressions
`srcref`	source reference.

Value

A list containing the parsed constituents

Author(s)

yihui

`parse.introduction` *Parse introduction: the premier part of a roxygen block containing description and option details separated by a blank roxygen line.*

Description

Copied from roxygen2: https://github.com/yihui/roxygen2/blob/master/R/parse-preref.R

Usage

```
parse.introduction(expression)
```

Arguments

`expression`	the description to be parsed

Value

A list containing the parsed description

Author(s)

yihui

`print.sqldoc` *Prints the SQL documentation.*

Description

Prints the SQL documentation.

Usage

```
## S3 method for class 'sqldoc'
print(x, ...)
```

Arguments

`x`	sqldoc object.
`...`	currently unused.

`recodeColumns` *Recodes factors with more than* `maxLevels` *to characters.*

Description

Recodes factors with more than `maxLevels` to characters.

Usage

```
recodeColumns(df, maxLevels = 20)
```

Arguments

`df`	the data frame to recode.
`maxLevels`	the maximum number of levels a factor can have before being converted to a character.

`sqldoc` *Parses the query and returns a list with all the elements of the comment.*

Description

Parses the query and returns a list with all the elements of the comment.

Usage

```
sqldoc(query)
```

Arguments

`query`	the query name.

Value

a list with documentation including `introduction`, `return`, and `params` (as a data frame).

`sqlexec` *Generic function for executing a query.*

Description

Generic function for executing a query.

Usage

```
sqlexec(connection, sql, ...)
```

Arguments

`connection`	the database connection.
`sql`	the query to execute.
`...`	other parameters passed to the appropriate `sqlexec` function.

Value

a data frame.

sqlexec.JDBCConnection *Executes queries for RJDBC*

Description

Executes queries for RJDBC

Usage

```
## S3 method for class 'JDBCConnection'
sqlexec(connection, sql, ...)
```

Arguments

`connection`	the database connection.
`sql`	the query to execute.
`...`	other parameters passed to the appropriate `sqlexec` function.

sqlexec.PostgreSQLConnection *Executes queries for RPostgreSQL*

Description

Executes queries for RPostgreSQL

Usage

```
## S3 method for class 'PostgreSQLConnection'
sqlexec(connection, sql, ...)
```

Arguments

`connection`	the database connection.
`sql`	the query to execute.
`...`	other parameters passed to the appropriate `sqlexec` function.

sqlexec.RMySQL *Executes queries for RMySQL package.*

Description

Executes queries for RMySQL package.

Usage

```
## S3 method for class 'RMySQL'
sqlexec(connection, sql, ...)
```

Arguments

`connection` the database connection.

`sql` the query to execute.

`...` other parameters passed to the appropriate `sqlexec` function.

sqlexec.RODBC *Executes queries for RODBC package.*

Description

Executes queries for RODBC package.

Usage

```
## S3 method for class 'RODBC'
sqlexec(connection, sql, ...)
```

Arguments

`connection` the database connection.

`sql` the query to execute.

`...` other parameters passed to the appropriate `sqlexec` function.

`sqlexec.SQLiteConnection` *Executes queries for RSQLite package.*

Description

Executes queries for RSQLite package.

Usage

```
## S3 method for class 'SQLiteConnection'
sqlexec(connection, sql, ...)
```

Arguments

`connection`	the database connection.
`sql`	the query to execute.
`...`	other parameters passed to the appropriate `sqlexec` function.

`sqlFile` *Returns the full path to the query or NULL if not found.*

Description

Returns the full path to the query or NULL if not found.

Usage

```
sqlFile(query)
```

Arguments

`query`	the query to find.

Value

path to the query file.

`sqlPaths` *Search paths for SQL repositories.*

Description

Search paths for SQL repositories.

Usage

```
sqlPaths(path, replace = TRUE)
```

Arguments

`path`	new path to add. This can a character vector of length greater than 1.
`replace`	if FALSE, the path(s) will be added to already existing list.

`sqlutils.envir` *The locations of SQL files*

Description

The locations of SQL files

Usage

```
sqlutils.envir
```

Format

```
<environment: 0x11220ce98>
```

Index