

Package ‘RSQL’

October 12, 2022

Type Package

Title Database Agnostic Package to Generate and Process 'SQL' Queries
in R

Version 0.2.1

Language en-US

Maintainer Alejandro Baranek <abaranek@dc.uba.ar>

Description

Allows the user to generate and execute select, insert, update and delete 'SQL' queries the underlying database without having to explicitly write 'SQL' code.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.2

Collate 'sql-lib.R' 'util-lib.R' 'zzz.R'

Imports lgr, R6, DBI

Suggests RSQLite, knitr, rmarkdown, dplyr, testthat, covr, lintr,
pkgdown

VignetteBuilder knitr

BugReports <https://github.com/rOpenStats/RSQL/issues>

URL <https://github.com/rOpenStats/RSQL>

NeedsCompilation no

Author Alejandro Baranek [cre, aut],
Leonardo Belen [aut]

Repository CRAN

Date/Publication 2022-05-21 06:20:02 UTC

R topics documented:

add_grep_exact_match	3
add_quotes	3

<code>cbind_coerced</code>	4
<code>check_fields_values</code>	4
<code>createRSQL</code>	5
<code>dequote</code>	5
<code>df_verify</code>	6
<code>genLogger</code>	6
<code>getLogger</code>	7
<code>getMtcarsdbPath</code>	7
<code>getPackageDir</code>	8
<code>is_quoted</code>	8
<code>needs_quotes</code>	9
<code>parse_where_clause</code>	9
<code>rename_col</code>	10
<code>replaceNAwithNULL</code>	10
<code>re_quote</code>	11
<code>re_quote_alt</code>	11
<code>rm_quotes</code>	12
<code>rm_vector_quotes</code>	12
<code>RSQL</code>	13
<code>RSQL.class</code>	13
<code>sql_execute_delete</code>	20
<code>sql_execute_get_insert</code>	21
<code>sql_execute_insert</code>	21
<code>sql_execute_select</code>	22
<code>sql_execute_update</code>	22
<code>sql_gen_delete</code>	23
<code>sql_gen_insert</code>	23
<code>sql_gen_select</code>	24
<code>sql_gen_update</code>	25
<code>sql_gen_where</code>	25
<code>sql_gen_where_list</code>	26
<code>sql_gen_where_or</code>	26
<code>sql_retrieve</code>	27
<code>sql_retrieve_insert</code>	27
<code>stuff_df_quoted</code>	28
<code>stuff_quote</code>	29
<code>trim</code>	29
<code>trim_leading</code>	30
<code>trim_trailing</code>	30
<code>%IN%</code>	31

`add_grep_exact_match` *add_grep_exact_match*

Description

`add_grep_exact_match`

Usage

`add_grep_exact_match(text)`

Arguments

text TEST

Author(s)

ken4rab

`add_quotes` *add_quotes*

Description

Adds quotes to a string

Usage

`add_quotes(text)`

Arguments

text The string to quote

Author(s)

ken4rab

cbind_coerced *cbind_coerced*

Description

cbind_coerced

Usage

```
cbind_coerced(...)
```

Arguments

... The parameters

Author(s)

ken4rab

check_fields_values *Check fields and values are sound*

Description

Check fields and values are sound

Usage

```
check_fields_values(fields, values, min.length = 0)
```

Arguments

fields Fields names to check
values values to check
min.length for vectors

Author(s)

ken4rab

createRSQL	<i>Produces a RSQL object</i>
------------	-------------------------------

Description

Produces a RSQL object

Usage

```
createRSQL(drv, dbname, user = NULL, password = NULL, host = NULL, port = NULL)
```

Arguments

drv	Driver name
dbname	Database name
user	Database user name
password	Database password
host	Database host
port	Database port

Author(s)

ken4rab

dequote	<i>Removes the quotes from the string</i>
---------	---

Description

Removes the quotes from the string

Usage

```
dequote(text)
```

Arguments

text	The string to remove the quotes from.
------	---------------------------------------

Author(s)

ken4rab

`df_verify`*Checks that the columns are in the data.frame*

Description

Checks that the columns are in the data.frame

Usage

```
df_verify(dataframe, columns)
```

Arguments

<code>dataframe</code>	The data.frame
<code>columns</code>	The columns to check

Author(s)

ken4rab

`genLogger`*genLogger*

Description

`genLogger`

Usage

```
genLogger(r6.object)
```

Arguments

<code>r6.object</code>	a R6 object with a logger member
------------------------	----------------------------------

Author(s)

ken4rab

`getLogger` *getLogger*

Description

`getLogger`

Usage

`getLogger(r6.object)`

Arguments

`r6.object` a R6 object with a logger member

Author(s)

ken4rab

`getMtcarsdbPath` *getCarsdbPath*

Description

`getCarsdbPath`

Usage

`getMtcarsdbPath(copy = TRUE)`

Arguments

`copy` a boolean that states whether it should be copied to the home directory or not.

Author(s)

ken4rab

getPackageDir	<i>Get package directory</i>
---------------	------------------------------

Description

Gets the path of package data.

Usage

```
getPackageDir()
```

Author(s)

ken4rab

is_quoted	<i>Determines if the string is quoted or not</i>
-----------	--

Description

Determines if the string is quoted or not

Usage

```
is_quoted(text, quotes_symbols = "'")
```

Arguments

text	The text to test
quotes_symbols	The quotation characters

Author(s)

ken4rab

needs_quotes	<i>Determines string type which needs quotes in an SQL statement</i>
--------------	--

Description

Determines string type which needs quotes in an SQL statement

Usage

```
needs_quotes(text)
```

Arguments

text	The text to test
------	------------------

Author(s)

ken4rab

parse_where_clause	<i>Parses a where clause.</i>
--------------------	-------------------------------

Description

Parses a where clause.

Usage

```
parse_where_clause(where_clause_list = c())
```

Arguments

where_clause_list	The list of params
-------------------	--------------------

Author(s)

ken4rab

<code>rename_col</code>	<i>rename_col</i>
-------------------------	-------------------

Description

renames a column on a data.frame

Usage

```
rename_col(df, name, replace_name)
```

Arguments

<code>df</code>	The data.frame
<code>name</code>	The name of the column
<code>replace_name</code>	The new name of the column

Author(s)

ken4rab

<code>replaceNAwithNULL</code>	<i>replaceNAwithNULL</i>
--------------------------------	--------------------------

Description

Replace NA with NULL in sql statement

Usage

```
replaceNAwithNULL(sql.code)
```

Arguments

<code>sql.code</code>	code to replace NA with NULL
-----------------------	------------------------------

Author(s)

ken4rab

re_quote	<i>This functions remove original quotes and sets validated quotes for corresponding db. If it had no quotes, will only put corresponding quotes symbols</i>
----------	--

Description

This functions remove original quotes and sets validated quotes for corresponding db. If it had no quotes, will only put corresponding quotes symbols

Usage

```
re_quote(text, quotes = "'')
```

Arguments

text	The string
quotes	The quotes

Author(s)

ken4rab

re_quote_alt	<i>This functions remove original quotes and sets validated quotes for corresponding db. If it had no quotes, will only put corresponding quotes symbols</i>
--------------	--

Description

This functions remove original quotes and sets validated quotes for corresponding db. If it had no quotes, will only put corresponding quotes symbols

Usage

```
re_quote_alt(text, quotes = "'')
```

Arguments

text	The string
quotes	The quotes

Author(s)

ken4rab

`rm_quotes`*rm_quotes*

Description

Removes quotes from the String

Usage

```
rm_quotes(text, quotes = "'")
```

Arguments

<code>text</code>	The string to remove quotes from
<code>quotes</code>	Quote characters

Author(s)

ken4rab

`rm_vector_quotes`*rm_vector_quotes*

Description

Removes quotes from data.frame columns

Usage

```
rm_vector_quotes(text.vector)
```

Arguments

<code>text.vector</code>	The text vector to remove quotes from.
--------------------------	--

Author(s)

ken4rab

RSQL	<i>rsql</i>
------	-------------

Description

A package to work with SQL datasources in a simple manner

Usage

```
.onLoad(libname, pkgname)
```

Arguments

libname	Library name
pkgname	Package name

Author(s)

Alejandro Baranek <abaranek@dc.uba.ar>, Leonardo Javier Belen <leobelen@gmail.com> Executes code while loading the package.

Examples

```
library(RSQL)
library(RSQLite)
db.name <- getMtcarsdbPath(copy = TRUE)
rsql <- createRSQL(drv = RSQLite::SQLite(), dbname = db.name)
select_sql <- rsql$gen_select(
  select_fields = "*", # c("wt", "qsec"),
  table = "mtcars",
  where_values = data.frame(carb = 8)
)
mtcars.observed <- rsql$execute_select(select_sql)
mtcars.observed
```

RSQL.class	<i>The class that provides the SQL functionality.</i>
------------	---

Description

This class is intended to simplify SQL commands.

Public fields

driver driver name
 db.name database name
 user db user
 password db password
 host db host
 port db port
 available.functions for generating select expressions
 entity.field.regexp for scrape a field or table expression
 entity.select.regexp for scrape a select expressions expression
 conn The connection handler
 valid.conn Checks if connection is valid
 last.query The last query
 last.rs The last resultset
 results.class Expected class for db results for running dbClearResult
 select.counter An instance select counter
 insert.counter An instance insert counter
 update.counter An instance update counter
 delete.counter An instance delete counter
 command.counter An instance command counter
 clear.rs.counter An instance clear.rs.counter
 logger is configued logger for current class

Methods**Public methods:**

- `RSQL.class$new()`
- `RSQL.class$connect()`
- `RSQL.class$setupResultClassFromDriver()`
- `RSQL.class$setupRegexp()`
- `RSQL.class$finalize()`
- `RSQL.class$checkEntitiesNames()`
- `RSQL.class$gen_select()`
- `RSQL.class$gen_insert()`
- `RSQL.class$gen_update()`
- `RSQL.class$gen_delete()`
- `RSQL.class$execute_select()`
- `RSQL.class$execute_update()`
- `RSQL.class$execute_insert()`
- `RSQL.class$execute_get_insert()`

- `RSQL.class$execute_command()`
- `RSQL.class$execute_delete()`
- `RSQL.class$retrieve()`
- `RSQL.class$retrieve_insert()`
- `RSQL.class$clearLastResult()`
- `RSQL.class$getSummary()`
- `RSQL.class$disconnect()`
- `RSQL.class$clone()`

Method `new()`: Initializes a connection

Usage:

```
RSQL.class$new(  
  drv,  
  dbname,  
  user = NULL,  
  password = NULL,  
  host = NULL,  
  port = NULL  
)
```

Arguments:

drv driver name
dbname database name
user user name
password password
host host name
port port number

Method `connect()`: Function which connects to database

Usage:

```
RSQL.class$connect()
```

Returns: conn object

Method `setupResultClassFromDriver()`: Infer ResultsClass from corresponding driver. Implemented for SQLiteDriver & PgConnection

Usage:

```
RSQL.class$setupResultClassFromDriver()
```

Returns: RSQL object

Method `setupRegexp()`: initialize regexp for scraping entities

Usage:

```
RSQL.class$setupRegexp(force = FALSE)
```

Arguments:

force force setup?

Returns: regexp for scraping select expressions

Method finalize(): Class destructor

Usage:

```
RSQL.class$finalize()
```

Method checkEntitiesNames(): Checks if an entity exists

Usage:

```
RSQL.class$checkEntitiesNames(entities, entity.type)
```

Arguments:

entities entities to check

entity.type entity type to check against

Method gen_select(): Generates a select

Usage:

```
RSQL.class$gen_select(
  select_fields,
  table,
  where_fields = names(where_values),
  where_values = NULL,
  group_by = c(),
  order_by = c(),
  top = 0,
  distinct = FALSE
)
```

Arguments:

select_fields fields to be selected

table table to select from

where_fields fields in the where clause

where_values values to the fields on the where clause

group_by fields to group by

order_by fields to order by

top where does the resultset starts?

distinct provides a way to select distinct rows

Method gen_insert(): Generate insert statement

Usage:

```
RSQL.class$gen_insert(table, values_df, insert_fields = names(values_df))
```

Arguments:

table The table to insert into

values_df The values to insert. Must be defined as data.frame of values

insert_fields the fields to insert into

Method gen_update(): Generate insert statement

Usage:

```
RSQL.class$gen_update(  
  table,  
  update_fields = names(values),  
  values,  
  where_fields = names(where_values),  
  where_values = NULL  
)
```

Arguments:

table the table to insert into
update_fields the fields to update
values the values to update
where_fields a where clause to the insert
where_values the values to add to the where clause

Method gen_delete(): Generate a delete statement

Usage:

```
RSQL.class$gen_delete(  
  table,  
  where_fields = names(where_values),  
  where_values = NULL  
)
```

Arguments:

table the table to insert into
where_fields a where clause to the insert
where_values the fields to add to the where clause

Method execute_select(): Performs an execution on the database

Usage:

```
RSQL.class$execute_select(sql_select)
```

Arguments:

sql_select the sql select statement to perform

Method execute_update(): Performs an update on the database

Usage:

```
RSQL.class$execute_update(sql_update)
```

Arguments:

sql_update the sql update statement to perform

Method execute_insert(): Performs an insert on the database

Usage:

```
RSQL.class$execute_insert(sql_insert)
```

Arguments:

sql_insert the sql insert statement to perform

Method execute_get_insert(): Performs a select statement, if not exists, executes insert statement

Usage:

```
RSQL.class$execute_get_insert(sql_select, sql_insert)
```

Arguments:

sql_select the sql select statement to perform

sql_insert the sql insert statement to perform

Method execute_command(): Performs a command on the database

Usage:

```
RSQL.class$execute_command(sql_command)
```

Arguments:

sql_command the sql statement to perform

Method execute_delete(): Performs an deletion on the database

Usage:

```
RSQL.class$execute_delete(sql_delete)
```

Arguments:

sql_delete the sql delete statement to perform

Method retrieve(): Performs an insert on the database. This is a composite function

Usage:

```
RSQL.class$retrieve(
  table,
  fields_uk = names(values_uk),
  values_uk,
  fields = names(values),
  values = NULL,
  field_id = "id"
)
```

Arguments:

table The table

fields_uk The fields unique key

values_uk The values unique key

fields The fields (Not used. Included for compatibility)

values The values (Not used. Included for compatibility)

field_id The field of the serial id

Method retrieve_insert(): Obtain id if object exists on the database. Insert object if not.

Usage:

```
RSQL.class$retrieve_insert(
  table,
  fields_uk = names(values_uk),
  values_uk,
  fields = names(values),
  values = data.frame(),
  field_id = "id"
)
```

Arguments:

table The table
 fields_uk The fields unique key
 values_uk The values unique key
 fields The fields
 values The values
 field_id The field of the serial id

Method clearLastResult(): clearLast Result for avoiding nasty warning getSummary

Usage:

```
RSQL.class$clearLastResult()
```

Method getSummary(): get RSQL summary string

Usage:

```
RSQL.class$getSummary()
```

Method disconnect(): Disconnects the instance from the database

Usage:

```
RSQL.class$disconnect()
```

Method clone(): The objects of this class are cloneable with this method.

Usage:

```
RSQL.class$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Author(s)

ken4rab

Examples

```
library(RSQL)
library(RSQLite)
db.name <- getMtcarsdbPath(copy = TRUE)
rsql <- createRSQL(drv = RSQLite::SQLite(), dbname = db.name)
select_sql <- rsql$gen_select(
  select_fields = "*", # c("wt", "qsec"),
```

```
    table = "mtcars",
    where_values = data.frame(carb = 8)
  )
mtcars.observed <- rsql$execute_select(select_sql)
mtcars.observed

mtcars.new <- mtcars.observed
mtcars.new$carb <- 9
insert_sql <- rsql$gen_insert(table = "mtcars", values_df = mtcars.new)
rsql$execute_insert(sql_insert = insert_sql)

where_values_df <- data.frame(carb = 9)
select_sql <- rsql$gen_select(
  select_fields = "*", # c("wt", "qsec"),
  table = "mtcars",
  where_values = data.frame(carb = 8)
)
mtcars.observed <- rsql$execute_select(select_sql)
mtcars.observed
```

sql_execute_delete *sql_execute_delete*

Description

Executes a delete on the Database

Usage

```
sql_execute_delete(sql_delete, dbconn = NULL)
```

Arguments

sql_delete	The delete SQL
dbconn	The Database Connection to run the query against

Author(s)

ken4rab

sql_execute_get_insert
sql_execute_get_insert

Description

Executes the insert statement

Usage

sql_execute_get_insert(dbconn, sql_select, sql_insert, ...)

Arguments

dbconn	The db connection
sql_select	The SQL select query
sql_insert	The SQL insert query
...	other variables to considered.

Author(s)

ken4rab

sql_execute_insert *Executes a statement on the database.*

Description

Executes a statement on the database.

Usage

sql_execute_insert(sql_insert, dbconn)

Arguments

sql_insert	The SQL String
dbconn	The Database Connection to run the query against

Author(s)

ken4rab

sql_execute_select *sql_execute_select*

Description

Executes a select on the database

Usage

sql_execute_select(sql_select, dbconn = NULL)

Arguments

sql_select The delete SQL
dbconn The Database Connection to run the query against

Author(s)

ken4rab

sql_execute_update *Executes an update on the database*

Description

Executes an update on the database

Usage

sql_execute_update(sql_update, dbconn = NULL)

Arguments

sql_update The update SQL
dbconn The Database Connection to run the query against

Author(s)

ken4rab

sql_gen_delete	<i>sql_gen_delete</i>
----------------	-----------------------

Description

Generates a Delete Statement

Usage

```
sql_gen_delete(table, where_fields = names(where_values), where_values = NULL)
```

Arguments

table	The table from which the delete statement will be generated
where_fields	The fields used in the where section
where_values	The values used in the where section

Author(s)

ken4rab

sql_gen_insert	<i>sql_gen_insert</i>
----------------	-----------------------

Description

Generates an insert statement.

Usage

```
sql_gen_insert(table, values_df, insert_fields = names(values_df))
```

Arguments

table	The table to be affected
values_df	The values to insert. Must be defined as data.frame of values
insert_fields	The fields to insert

Author(s)

ken4rab

sql_gen_select	<i>sql_gen_select</i>
----------------	-----------------------

Description

Generates a Select Statement

Usage

```
sql_gen_select(  
  select_fields,  
  table,  
  where_fields = names(where_values),  
  where_values = NULL,  
  group_by = c(),  
  order_by = c(),  
  top = 0,  
  distinct = FALSE  
)
```

Arguments

select_fields	The fields to be selected
table	The table to be used in the select
where_fields	The fields used in the where section
where_values	The values used in the where section
group_by	Group by fields
order_by	Order by fields
top	Retrieve top records
distinct	it adds a distinct clause to the query.

Author(s)

ken4rab

sql_gen_update	<i>sql_gen_update</i>
----------------	-----------------------

Description

Generates an update statement

Usage

```
sql_gen_update(
    table,
    update_fields = names(values),
    values,
    where_fields = names(where_values),
    where_values
)
```

Arguments

table	The table to update
update_fields	The fields to update
values	The values to update
where_fields	The fields for where statement
where_values	The values for where statement

Author(s)

ken4rab

sql_gen_where	<i>sql_gen_where</i>
---------------	----------------------

Description

Generates a where statement to be used on a SQL statement.

Usage

```
sql_gen_where(where_fields = names(where_values), where_values)
```

Arguments

where_fields	The fields used in the where section
where_values	The values used in the where section

Author(s)

ken4rab

`sql_gen_where_list` *sql_gen_where_list*

Description

Generates a where list statement to be used on a SQL statement.

Usage

```
sql_gen_where_list(where_fields, where_values)
```

Arguments

`where_fields` The fields used in the where section

`where_values` The values used in the where section

Author(s)

ken4rab

`sql_gen_where_or` *sql_gen_where_or*

Description

Generates a where (or) statement to be used on a SQL statement.

Usage

```
sql_gen_where_or(where_fields = names(where_values), where_values)
```

Arguments

`where_fields` The fields used in the where section

`where_values` The values used in the where section

Author(s)

ken4rab

sql_retrieve	<i>Retrieves Statement</i>
--------------	----------------------------

Description

Retrieves Statement

Usage

```
sql_retrieve(
    table,
    fields_uk = names(values_uk),
    values_uk,
    fields = names(values),
    values = NULL,
    field_id = "id",
    dbconn = NULL
)
```

Arguments

table	The table
fields_uk	The fields unique key
values_uk	The values unique key
fields	The fields (Not used. Included for compatibility)
values	The values (Not used. Included for compatibility)
field_id	The field of the serial id
dbconn	The database connection

Author(s)

ken4rab

sql_retrieve_insert	<i>Retrieves or insert Statement</i>
---------------------	--------------------------------------

Description

Retrieves or insert Statement

Usage

```
sql_retrieve_insert(
  table,
  fields_uk = names(values_uk),
  values_uk,
  fields = names(values),
  values = NULL,
  field_id = "id",
  dbconn = NULL
)
```

Arguments

table	The table
fields_uk	The fields unique key
values_uk	The values unique key
fields	The fields
values	The values
field_id	The field of the serial id
dbconn	The database connection

Author(s)

ken4rab

stuff_df_quoted	<i>stuff_df_quoted</i>
-----------------	------------------------

Description

stuff quote characters in quoted or not quoted df for DSL or DML operations

Usage

```
stuff_df_quoted(text.df)
```

Arguments

text.df	Data Frame with corresponding values and fields as colnames
---------	---

Author(s)

ken4rab

stuff_quote	<i>Stuff quote symbol from text</i>
-------------	-------------------------------------

Description

Stuff quote symbol from text

Usage

```
stuff_quote(unquoted.text, quote = "'")
```

Arguments

unquoted.text	The unquoted string to stuff quotes from.
quote	The quoting symbol. Default is ' .

Author(s)

ken4rab

trim	<i>Returns string w/o leading or trailing whitespace</i>
------	--

Description

Returns string w/o leading or trailing whitespace

Usage

```
trim(x)
```

Arguments

x	The string
---	------------

Author(s)

ken4rab

<code>trim_leading</code>	<i>trim_leading</i>
---------------------------	---------------------

Description

Returns string w/o leading whitespace

Usage

```
trim_leading(x)
```

Arguments

x	The string
---	------------

Author(s)

ken4rab

<code>trim_trailing</code>	<i>trim_trailing</i>
----------------------------	----------------------

Description

Returns string w/o trailing whitespace

Usage

```
trim_trailing(x)
```

Arguments

x	The string
---	------------

Author(s)

ken4rab

%IN% *Operator IN for multiple columns*

Description

Operator IN for multiple columns

Usage

x %IN% y

Arguments

x vector x

y vector y

Author(s)

ken4rab

Index

.onLoad (RSQL), 13
%IN%, 31

add_grep_exact_match, 3
add_quotes, 3

cbind_coerced, 4
check_fields_values, 4
createRSQL, 5

dequote, 5
df_verify, 6

genLogger, 6
getLogger, 7
getMtcarsdbPath, 7
getPackageDir, 8

is_quoted, 8

needs_quotes, 9

parse_where_clause, 9

re_quote, 11
re_quote_alt, 11
rename_col, 10
replaceNAwithNULL, 10
rm_quotes, 12
rm_vector_quotes, 12
RSQL, 13
RSQL.class, 13

sql_execute_delete, 20
sql_execute_get_insert, 21
sql_execute_insert, 21
sql_execute_select, 22
sql_execute_update, 22
sql_gen_delete, 23
sql_gen_insert, 23
sql_gen_select, 24
sql_gen_update, 25
sql_gen_where, 25
sql_gen_where_list, 26
sql_gen_where_or, 26
sql_retrieve, 27
sql_retrieve_insert, 27
stuff_df_quoted, 28
stuff_quote, 29

trim, 29
trim_leading, 30
trim_trailing, 30