Package 'sqliter'

May 9, 2026

Type Package

Title Connection wrapper to SQLite databases

Version 0.1.0

Author Wilson Freitas <wilson.freitas@gmail.com>

Maintainer Wilson Freitas <wilson.freitas@gmail.com>

URL https://github.com/wilsonfreitas/sqliter/

Description sqliter helps users, mainly data munging practioneers, to organize their sql calls in a clean structure. It simplifies the process of extracting and transforming data into useful formats.

License MIT + file LICENSE

Imports stringr, functional, DBI, RSQLite

Collate 'sqliter.R'

NeedsCompilation no

Repository CRAN

Date/Publication 2014-01-26 19:54:31

Contents

sqliter-package *Functions to wrap SQLite calls*

Description

sqliter helps users, mainly data munging practioneers, to organize their sql calls in a clean structure. It simplifies the process of extracting and transforming data into useful formats.

execute *execute query into a given database*

Description

Once you have a `sqliter` database properly set you can execute queries into that database and get your data transformed. By default this function returns a data.frame object, but if you transform your data you can get whatever you need.

Usage

```
execute(object, ...)

## S3 method for class 'sqliter'
execute(object, database, query,
  post_proc = identity, ...)
```

Arguments

`object`	`sqliter` object
`database`	the SQLite database filename without extension
`query`	the query string
`post_proc`	a function to transform data, it receives a database and returns whatever you need.
`...`	additional arguments used by prepared queries

Examples

```
## Not run:
DBM <- sqliter(path=c("data", "another/project/data"))
ds <- execute(DBM, "dummydatabase", "select count(*) from dummytable")
ds <- execute(DBM, "dummydatabase", "select * from dummytable where
      name = :name", name=c("Macunamima", "Borba Gato"))
ds <- execute(DBM, "dummydatabase", "select * from dummytable where
      name = :name", name=c("Macunamima", "Borba Gato"),
        post_proc=function(ds) {
ds <- transform(ds, birthday=as.Date(birthday))
```

```
ds
})

## End(Not run)
```

`find_database` *returns the paths of the given database*

Description

returns the paths of the given database

Usage

```
find_database(object, database)

## S3 method for class 'sqliter'
find_database(object, database)
```

Arguments

`object`	`sqliter` object
`database`	the SQLite database filename without extension

Examples

```
## Not run:
DBM <- sqliter(path=c("data", "another/project/data"))
find_database(DBM, "dummydatabase")
# "data/dummydatabase.db"

## End(Not run)
```

`query-functions` *query functions*

Description

query functions are dynamic functions which connect to a database, execute queries in it and transform data. Actually it is a decorator for `execute` function. `execute` has 5 arguments. The first argument is an instance of the `sqliter` class and the second is the database name. The call to a query function is executed like a method call to the `sqliter` object through the $ operator. The function name must have the following pattern: `query_<database name without extension>`. This call returns an `execute` function with the first 2 argument already set. The first parameter is the `sqliter` object on which the $ operator have been called and the second argument is extracted from the query function name, the name after the preffix `query_`.

Examples

```
## Not run:
DBM <- sqliter(path=c("data", "another/project/data"))
DBM$query_dummydatabase("select count(*) from dummytable")

## End(Not run)
```

`sqliter` *Creates the sqliter a kinf of SQLite database manager, but not that far.*

Description

`sqliter` object works pretty much like a database manager helping users to execute queries and transform data through a clean interface.

Usage

```
sqliter(...)
```

Arguments

`...`	arguments such as `path` must be provided during object instantiation.

Examples

```
## Not run: DBM <- sqliter(path=c("data", "another/project/data"))
```

Index