Package ‘readwritesqlite’

October 16, 2022

Title Enhanced Reading and Writing for 'SQLite' Databases

Version 0.2.0

Description Reads and writes data frames to 'SQLite' databases while preserving time zones (for POSIXct columns), projections (for 'sfc' columns), units (for 'units' columns), levels (for factors and ordered factors) and classes for logical, Date and 'hms' columns. It also logs changes to tables and provides more informative error messages.

License MIT + file LICENSE

URL `https://github.com/poissonconsulting/readwritesqlite`

BugReports `https://github.com/poissonconsulting/readwritesqlite/issues`

Depends R (>= 4.0)

Imports chk, DBI, hms, lifecycle, RSQLite, tibble, sf, rlang, glue, crayon

Suggests covr, knitr, pool, rmarkdown, testthat (>= 3.0.0), withr, units

VignetteBuilder knitr

RdMacros lifecycle

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.2.1

Config/testthat/edition 3

NeedsCompilation no

Author Joe Thorley [aut, cre] (<`https://orcid.org/0000-0002-7683-4592`>), Sebastian Dalgarno [ctb] (<`https://orcid.org/0000-0002-3658-4517`>), Poisson Consulting [cph, fnd]

Maintainer Joe Thorley `<joe@poissonconsulting.ca>`

Repository CRAN

Date/Publication 2022-10-16 19:40:02 UTC

R topics documented:

`chk_sqlite_conn` *Check SQLite Connection*

Description

`chk_sqlite_conn` checks if a SQLite connection.

Usage

```
chk_sqlite_conn(x, connected = NA, x_name = NULL)

check_sqlite_connection(
  x,
  connected = NA,
  x_name = substitute(x),
  error = TRUE
)
```

Arguments

`x`	The object to check.
`connected`	A logical scalar specifying whether x should be connected.
`x_name`	A string of the name of object x or NULL.
`error`	A flag specifying whether to through an error if the check fails.

Value

`NULL`, invisibly. Called for the side effect of throwing an error if the condition is not met.

Functions

- `check_sqlite_connection()`: Check SQLite Connection

Examples

```
conn <- rws_connect()
chk_sqlite_conn(conn)
rws_disconnect(conn)
try(chk_sqlite_conn(conn, connected = TRUE))
```

`rws_connect` *Opens SQLite Database Connection*

Description

Opens a SQLiteConnection to a SQLite database with foreign key constraints enabled.

Usage

```
rws_connect(dbname = ":memory:", exists = NA)
```

Arguments

`dbname` The path to the database file. SQLite keeps each database instance in one single file. The name of the database *is* the file name, thus database names should be legal file names in the running platform. There are two exceptions:

- `""` will create a temporary on-disk database. The file will be deleted when the connection is closed.
- `":memory:"` or `"file::memory:"` will create a temporary in-memory database.

`exists` A flag specifying whether the table(s) must already exist.

Value

A SQLiteConnection to a SQLite database with foreign key constraints enabled.

See Also

`rws_disconnect()`

Examples

```
conn <- rws_connect()
print(conn)
rws_disconnect(conn)
```

`rws_data` *Example Data*

Description

An sf tibble of example data.

Usage

```
rws_data
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 3 rows and 6 columns.

Examples

```
rws_data
```

`rws_describe_meta` *Add Descriptions to SQL Meta Data Table*

Description

Add Descriptions to SQL Meta Data Table

Usage

```
rws_describe_meta(x, ..., conn)
```

Arguments

`x`	An object specifying the descriptions.
`...`	Not used.
`conn`	A SQLiteConnection to a database.

Value

An invisible copy of the updated meta table.

See Also

Other rws_describe_meta: `rws_describe_meta.character()`

`rws_describe_meta.character` *Add Descriptions to SQL Meta Data Table*

Description

Add Descriptions to SQL Meta Data Table

Usage

```
## S3 method for class 'character'
rws_describe_meta(x, column, description, ..., conn)
```

Arguments

`x`	A character vector of table name(s).
`column`	A character vector of column name(s).
`description`	A character vector of the description(s)
`...`	Not used.
`conn`	A SQLiteConnection to a database.

Value

An invisible copy of the updated meta table.

See Also

Other rws_describe_meta: `rws_describe_meta()`

Examples

```
conn <- rws_connect()
rws_write(rws_data, exists = FALSE, conn = conn)
rws_read_meta(conn)
rws_describe_meta("rws_data", "Units", "The site length.", conn = conn)
rws_describe_meta("rws_data", "POSIXct", "Time of the visit", conn = conn)
rws_read_meta(conn)
rws_disconnect(conn)
```

`rws_describe_meta.data.frame` *Add Data Frame of Descriptions to SQL Meta Data Table*

Description

Add Data Frame of Descriptions to SQL Meta Data Table

Usage

```
## S3 method for class 'data.frame'
rws_describe_meta(x, ..., conn)
```

Arguments

`x`	A data frame with columns Table, Column, Description.
`...`	Not used.
`conn`	A SQLiteConnection to a database.

Value

An invisible character vector of the previous descriptions.

See Also

Other rws_read: `rws_read.SQLiteConnection()`, `rws_read.character()`, `rws_read()`

`rws_disconnect` *Close SQLite Database Connection*

Description

Closes a SQLiteConnection to a SQLite database.

Usage

```
rws_disconnect(conn)
```

Arguments

`conn`	An `RSQLite::SQLiteConnection()`.

See Also

`rws_connect()`

Examples

```
conn <- rws_connect()
rws_disconnect(conn)
print(conn)
```

`rws_drop_table` *Drop SQLite Table*

Description

Drops SQLite table using DROP TABLE.

Usage

```
rws_drop_table(table_name, conn)
```

Arguments

`table_name`	A string of the name of the table.
`conn`	A SQLiteConnection to a database.

Details

Also drops rows from meta and init tables.

Value

TRUE

References

`https://www.sqlite.org/lang_droptable.html`

See Also

Other rws_rename: `rws_rename_column()`, `rws_rename_table()`

Examples

```
conn <- rws_connect()
rws_write(rws_data, exists = FALSE, conn = conn)
rws_list_tables(conn)
rws_drop_table("rws_data", conn = conn)
rws_list_tables(conn)
rws_disconnect(conn)
```

`rws_export_gpkg` *Export all spatial datasets in a database as geopackages.*

Description

Export all spatial datasets in a database as geopackages.

Usage

```
rws_export_gpkg(conn, dir, overwrite = FALSE)
```

Arguments

`conn`	A SQLiteConnection to a database.
`dir`	A string of the path to the directory to save the geopackages in.
`overwrite`	A flag specifying whether to overwrite existing geopackages.

Details

If more than one spatial column is present in a table, a separate geopackage will be exported for each, and the other spatial columns will be dropped.

Value

An invisible named vector of the file names and new file names saved.

`rws_list_tables` *Table Names*

Description

Gets the table names excluding the names of the meta and log tables.

Usage

```
rws_list_tables(conn)
```

Arguments

`conn`	A SQLiteConnection to a database.

Value

A character vector of table names.

Examples

```
conn <- rws_connect()
rws_list_tables(conn)
rws_write(rws_data, exists = FALSE, conn = conn)
rws_list_tables(conn)
rws_disconnect(conn)
```

`rws_query` *Query SQLite Database*

Description

Gets a query from a SQLite database.

Usage

```
rws_query(query, meta = TRUE, conn)
```

Arguments

`query`	A string of a SQLite query.
`meta`	A flag specifying whether to preserve meta data.
`conn`	A SQLiteConnection to a database.

Value

A data frame of the query.

Examples

```
conn <- rws_connect()
rws_write(rws_data, exists = FALSE, conn = conn)
rws_query("SELECT date, posixct, factor FROM rws_data", conn = conn)
rws_disconnect(conn)
```

`rws_read` *Read from a SQLite Database*

Description

Read from a SQLite Database

Usage

```
rws_read(x, ...)
```

Arguments

`x`	An object specifying the table(s) to read.
`...`	Not used.

Value

A named list of data frames.

See Also

Other rws_read: `rws_describe_meta.data.frame()`, `rws_read.SQLiteConnection()`, `rws_read.character()`

`rws_read.character` *Read Tables from a SQLite Database*

Description

Read Tables from a SQLite Database

Usage

```
## S3 method for class 'character'
rws_read(x, meta = TRUE, conn, ...)
```

Arguments

`x`	A character vector of table names.
`meta`	A flag specifying whether to preserve meta data.
`conn`	A SQLiteConnection to a database.
`...`	Not used.

Value

A named list of the data frames.

See Also

Other rws_read: `rws_describe_meta.data.frame()`, `rws_read.SQLiteConnection()`, `rws_read()`

Examples

```
conn <- rws_connect()
rws_write(rws_data, exists = FALSE, conn = conn)
rws_write(rws_data[c("date", "ordered")],
  x_name = "data2",
  exists = FALSE, conn = conn
)
rws_read(c("rws_data", "data2"), conn = conn)
rws_disconnect(conn)
```

`rws_read.SQLiteConnection` *Read All Tables from a SQLite Database*

Description

Read All Tables from a SQLite Database

Usage

```
## S3 method for class 'SQLiteConnection'
rws_read(x, meta = TRUE, ...)
```

Arguments

`x`	A SQLiteConnection to a database.
`meta`	A flag specifying whether to preserve meta data.
`...`	Not used.

Value

A named list of the data frames.

See Also

Other rws_read: `rws_describe_meta.data.frame()`, `rws_read.character()`, `rws_read()`

Examples

```
conn <- rws_connect()
rws_write(rws_data, exists = FALSE, conn = conn)
rws_write(rws_data[c("date", "ordered")],
  x_name = "data2", exists = FALSE, conn = conn
)
rws_read(conn)
rws_disconnect(conn)
```

`rws_read_init` *Read Initialization Data table from a SQLite Database*

Description

The table is created if it doesn't exist.

Usage

```
rws_read_init(conn)
```

Arguments

`conn`	A SQLiteConnection to a database.

Value

A data frame of the init table

Examples

```
conn <- rws_connect()
rws_read_init(conn)
rws_write(rws_data, exists = FALSE, conn = conn)
rws_read_init(conn)
rws_disconnect(conn)
```

`rws_read_log` *Read Log Data Table from a SQLite Database*

Description

The table is created if it doesn't exist.

Usage

```
rws_read_log(conn)
```

Arguments

`conn`	A SQLiteConnection to a database.

Value

A data frame of the log table

Examples

```
conn <- rws_connect()
rws_read_log(conn)
rws_write(rws_data, exists = FALSE, conn = conn)
## Not run:
rws_read_log(conn)

## End(Not run)
rws_disconnect(conn)
```

`rws_read_meta` *Read Meta Data table from a SQLite Database*

Description

The table is created if it doesn't exist.

Usage

```
rws_read_meta(conn)
```

Arguments

`conn`	A SQLiteConnection to a database.

Value

A data frame of the meta table

Examples

```
conn <- rws_connect()
rws_read_meta(conn)
rws_write(rws_data, exists = FALSE, conn = conn)
rws_read_meta(conn)
rws_disconnect(conn)
```

rws_read_table *Read a Table from a SQLite Database*

Description

Read a Table from a SQLite Database

Usage

```
rws_read_table(x, meta = TRUE, conn)
```

Arguments

x	A string of the table name.
meta	A flag specifying whether to preserve meta data.
conn	A SQLiteConnection to a database.

Value

A data frame of the table.

Examples

```
conn <- rws_connect()
rws_write(rws_data, exists = FALSE, conn = conn)
rws_write(rws_data[c("date", "ordered")],
  x_name = "data2", exists = FALSE, conn = conn
)
rws_read_table("data2", conn = conn)
rws_disconnect(conn)
```

rws_rename_column *Rename SQLite Column*

Description

Rename SQLite Column

Usage

```
rws_rename_column(table_name, column_name, new_column_name, conn)
```

Arguments

`table_name`	A string of the name of the table.
`column_name`	A string of the column name.
`new_column_name`	A string of the new name for the column.
`conn`	A SQLiteConnection to a database.

Value

TRUE

See Also

Other rws_rename: `rws_drop_table()`, `rws_rename_table()`

Examples

```
conn <- rws_connect()
rws_write(data.frame(x = 1), x_name = "local", exists = FALSE, conn = conn)
rws_read_table("local", conn = conn)
rws_rename_column("local", "x", "Y", conn = conn)
rws_read_table("local", conn = conn)
rws_disconnect(conn)
```

`rws_rename_table` *Rename SQLite Table*

Description

Rename SQLite Table

Usage

```
rws_rename_table(table_name, new_table_name, conn)
```

Arguments

`table_name`	A string of the name of the table.
`new_table_name`	A string of the new name for the table.
`conn`	A SQLiteConnection to a database.

Value

TRUE

See Also

Other rws_rename: `rws_drop_table()`, `rws_rename_column()`

Examples

```
conn <- rws_connect()
rws_write(rws_data, exists = FALSE, conn = conn)
rws_list_tables(conn)
rws_rename_table("rws_data", "tableb", conn)
rws_list_tables(conn)
rws_disconnect(conn)
```

`rws_write` *Write to a SQLite Database*

Description

Write to a SQLite Database

Usage

```
rws_write(
  x,
  exists = TRUE,
  delete = FALSE,
  replace = FALSE,
  meta = TRUE,
  log = TRUE,
  commit = TRUE,
  strict = TRUE,
  x_name = substitute(x),
  silent = getOption("rws.silent", FALSE),
  conn,
  ...
)
```

Arguments

`x`	The object to write.
`exists`	A flag specifying whether the table(s) must already exist.
`delete`	A flag specifying whether to delete existing rows before inserting data. If `meta = TRUE` the meta data is deleted.
`replace`	A flag specifying whether to replace any existing rows whose inclusion would violate unique or primary key constraints.
`meta`	A flag specifying whether to preserve meta data.
`log`	A flag specifying whether to log the table operations.

`commit`	A flag specifying whether to commit the operations (calling with commit = FALSE can be useful for checking data).
`strict`	A flag specifying whether to error if x has extraneous columns or if exists = TRUE extraneous data frames.
`x_name`	A string of the name of the object.
`silent`	A flag specifying whether to suppress messages and warnings.
`conn`	A SQLiteConnection to a database.
`...`	Not used.

Value

An invisible character vector of the name(s) of the table(s).

See Also

Other rws_write: `rws_write.data.frame()`, `rws_write.environment()`, `rws_write.list()`

Examples

```
conn <- rws_connect()
rws_write(rws_data, exists = FALSE, conn = conn)
rws_disconnect(conn)
```

`rws_write.data.frame` *Write a Data Frame to a SQLite Database*

Description

Write a Data Frame to a SQLite Database

Usage

```
## S3 method for class 'data.frame'
rws_write(
  x,
  exists = TRUE,
  delete = FALSE,
  replace = FALSE,
  meta = TRUE,
  log = TRUE,
  commit = TRUE,
  strict = TRUE,
  x_name = substitute(x),
  silent = getOption("rws.silent", FALSE),
  conn,
  ...
)
```

Arguments

`x`	A data frame.
`exists`	A flag specifying whether the table(s) must already exist.
`delete`	A flag specifying whether to delete existing rows before inserting data. If `meta = TRUE` the meta data is deleted.
`replace`	A flag specifying whether to replace any existing rows whose inclusion would violate unique or primary key constraints.
`meta`	A flag specifying whether to preserve meta data.
`log`	A flag specifying whether to log the table operations.
`commit`	A flag specifying whether to commit the operations (calling with commit = FALSE can be useful for checking data).
`strict`	A flag specifying whether to error if x has extraneous columns or if exists = TRUE extraneous data frames.
`x_name`	A string of the name of the object.
`silent`	A flag specifying whether to suppress messages and warnings.
`conn`	A SQLiteConnection to a database.
`...`	Not used.

See Also

Other rws_write: `rws_write.environment()`, `rws_write.list()`, `rws_write()`

Examples

```
conn <- rws_connect()
rws_list_tables(conn)
rws_write(rws_data, exists = FALSE, conn = conn)
rws_write(rws_data, x_name = "moredata", exists = FALSE, conn = conn)
rws_list_tables(conn)
rws_disconnect(conn)
```

`rws_write.environment` *Write the Data Frames in an Environment to a SQLite Database*

Description

Write the Data Frames in an Environment to a SQLite Database

Usage

```
## S3 method for class 'environment'
rws_write(
  x,
  exists = TRUE,
  delete = FALSE,
  replace = FALSE,
  meta = TRUE,
  log = TRUE,
  commit = TRUE,
  strict = TRUE,
  x_name = substitute(x),
  silent = getOption("rws.silent", FALSE),
  conn,
  all = TRUE,
  unique = TRUE,
  ...
)
```

Arguments

`x`	An environment.
`exists`	A flag specifying whether the table(s) must already exist.
`delete`	A flag specifying whether to delete existing rows before inserting data. If `meta = TRUE` the meta data is deleted.
`replace`	A flag specifying whether to replace any existing rows whose inclusion would violate unique or primary key constraints.
`meta`	A flag specifying whether to preserve meta data.
`log`	A flag specifying whether to log the table operations.
`commit`	A flag specifying whether to commit the operations (calling with commit = FALSE can be useful for checking data).
`strict`	A flag specifying whether to error if x has extraneous columns or if exists = TRUE extraneous data frames.
`x_name`	A string of the name of the object.
`silent`	A flag specifying whether to suppress messages and warnings.
`conn`	A SQLiteConnection to a database.
`all`	A flag specifying whether all the existing tables in the data base must be represented.
`unique`	A flag specifying whether each table must represented by no more than one data frame.
`...`	Not used.

See Also

Other rws_write: `rws_write.data.frame()`, `rws_write.list()`, `rws_write()`

Examples

```
conn <- rws_connect()
rws_list_tables(conn)
atable <- readwritesqlite::rws_data
another_table <- readwritesqlite::rws_data
not_atable <- 1L
rws_write(environment(), exists = FALSE, conn = conn)
rws_list_tables(conn)
rws_disconnect(conn)
```

rws_write.list *Write a Named List of Data Frames to a SQLite Database*

Description

Write a Named List of Data Frames to a SQLite Database

Usage

```
## S3 method for class 'list'
rws_write(
  x,
  exists = TRUE,
  delete = FALSE,
  replace = FALSE,
  meta = TRUE,
  log = TRUE,
  commit = TRUE,
  strict = TRUE,
  x_name = substitute(x),
  silent = getOption("rws.silent", FALSE),
  conn,
  all = TRUE,
  unique = TRUE,
  ...
)
```

Arguments

`x`	A named list of data frames.
`exists`	A flag specifying whether the table(s) must already exist.
`delete`	A flag specifying whether to delete existing rows before inserting data. If `meta = TRUE` the meta data is deleted.
`replace`	A flag specifying whether to replace any existing rows whose inclusion would violate unique or primary key constraints.
`meta`	A flag specifying whether to preserve meta data.

`log`	A flag specifying whether to log the table operations.
`commit`	A flag specifying whether to commit the operations (calling with commit = FALSE can be useful for checking data).
`strict`	A flag specifying whether to error if x has extraneous columns or if exists = TRUE extraneous data frames.
`x_name`	A string of the name of the object.
`silent`	A flag specifying whether to suppress messages and warnings.
`conn`	A SQLiteConnection to a database.
`all`	A flag specifying whether all the existing tables in the data base must be represented.
`unique`	A flag specifying whether each table must represented by no more than one data frame.
`...`	Not used.

See Also

Other rws_write: `rws_write.data.frame()`, `rws_write.environment()`, `rws_write()`

Examples

```
conn <- rws_connect()
rws_list_tables(conn)
rws_write(list(somedata = rws_data, anothertable = rws_data), exists = FALSE, conn = conn)
rws_list_tables(conn)
rws_disconnect(conn)
```

`vld_sqlite_conn` *Validate SQLite Connection*

Description

Validate SQLite Connection

Usage

```
vld_sqlite_conn(x, connected = NA)
```

Arguments

`x`	The object to check.
`connected`	A logical scalar specifying whether x should be connected.

Value

A flag indicating whether the object was validated.

Examples

```
conn <- rws_connect()
vld_sqlite_conn(conn)
rws_disconnect(conn)
vld_sqlite_conn(conn, connected = TRUE)
```

Index