

Package ‘frostlake’

October 14, 2026

Title 'DBI' Backend for the 'Frostlake' Database Engine

Version 0.1.0

Description A 'DBI'-compliant driver for 'Frostlake', a SQL engine emulating core 'Snowflake' features. Speaks the engine's HTTP protocol against a running 'DatabaseHttpServer': one POST per statement with a session id the server issues on first contact, so session state (current database/schema, transactions) persists across statements. Query results arrive as data frames with Snowflake types mapped to R types.

License Apache License (== 2.0)

URL <https://github.com/Frostlake-DB/frostlake-r>

BugReports <https://github.com/Frostlake-DB/frostlake-r/issues>

Encoding UTF-8

Depends R (>= 4.1)

Imports DBI (>= 1.1.0), curl, jsonlite, methods

Suggests testthat (>= 3.0.0), withr

Config/testthat/edition 3

Collate 'utils.R' 'substitute.R' 'wire.R' 'driver.R' 'connection.R'
'result.R' 'tables.R'

NeedsCompilation no

Author MLorek [aut, cre, cph]

Maintainer MLorek <michal.lorek@gmail.com>

Repository CRAN

Date/Publication 2026-10-14 10:50:02 UTC

Contents

frostlake	2
frostlake-methods	3
Index	5

frostlake*DBI driver for the Frostlake database engine*

Description

frostlake() returns the DBI driver object. Hand it to [dbConnect](#) together with the coordinates of a running Frostlake DatabaseHttpServer; every DBI verb then works on the connection. One HTTP POST per statement carries the SQL and a session id the server issues on first contact, so session state (current database/schema, transactions) persists across statements.

Usage

```
frostlake()
```

Value

A FrostlakeDriver object for [dbConnect](#).

Examples

```
drv <- frostlake()
drv

# The engine types dbWriteTable() creates columns with, for each R type
DBI::dbDataType(drv, data.frame(id = 1:2, name = c("a", "b"), day = Sys.Date()))

## Not run:
# Everything below needs a running Frostlake DatabaseHttpServer
library(DBI)
con <- dbConnect(frostlake::frostlake(),
                 host = "localhost", port = 18082,
                 dbname = "MY_DB", schema = "PUBLIC")
# or: dbConnect(frostlake::frostlake(), url = "frostlake://localhost:18082/MY_DB?schema=PUBLIC")

dbExecute(con, "CREATE TABLE people (id INTEGER, name VARCHAR)")
dbExecute(con, "INSERT INTO people VALUES (?, ?)", params = list(1L, "Ada"))
dbGetQuery(con, "SELECT id, name FROM people WHERE id = ?", params = list(1L))

dbWriteTable(con, "SCORES", data.frame(id = 1:3, score = c(9.5, 8, 7)))
dbReadTable(con, "SCORES")
dbDisconnect(con)

## End(Not run)
```

Description

Implementations of the standard **DBI** generics for `FrostlakeDriver`, `FrostlakeConnection` and `FrostlakeResult`. Each follows its generic’s contract — see the corresponding **DBI** help page — with these engine-specific notes:

- `dbConnect` applies the DSN’s database and schema eagerly with `USE DATABASE/USE SCHEMA`, so an unknown database fails at connect. A plain name travels bare whatever its case and resolves case-insensitively, as on Snowflake; wrap the value in double quotes itself for an exact-case match.
- Every request after the first names the connection’s session, and once the engine reports `newSession` (0.1.0 and later do) it also sends `requireSession`, so a session the engine no longer holds is refused instead of being replaced by a fresh one. The statement did not run: when the lost session held nothing of its own, a fresh session takes the connection’s database and schema and the statement is sent once more; when it held a transaction or context (`USE`, `SET`, `ALTER SESSION`, a temporary object), that raises `frostlake_session_lost`, a subclass of `frostlake_error`, and the next statement starts afresh on the connection’s scope. `dbDisconnect` releases the session with `DELETE /api/sessions/{id}`, which rolls back a transaction left open; against an older engine it sends a `ROLLBACK` instead.
- `dbSendQuery/dbGetQuery/dbExecute` accept params: unnamed elements fill `?` placeholders positionally, named elements fill `@name/:name` placeholders, all inlined client-side. Placeholders inside string literals, quoted identifiers, `$$...$$` bodies and comments are left untouched; `::` is always the cast operator. A positional value `no ?` consumed is an error. With placeholders and no params, execution waits for `dbBind`, which may carry equal-length vectors and runs the statement once per index. `multi_statement_count` declares how many statements the text carries (0 for any number) for engines that enforce the session’s statement count; queries materialize the first result set and affected-row counts sum across all of them. A Snowflake Scripting block (text opening with `DECLARE`, or with a `BEGIN` that starts a block) owns its `?` placeholders, which reach the engine untouched. An empty statement is sent too, for the engine to refuse. A refusal the engine reports raises `frostlake_sql_error`, a subclass of `frostlake_error`.
- `dbExecute` and `dbGetRowsAffected` use the affected-row count engines from 0.1.0 report per result. Against an older engine they sum every “number of rows ...” column of a single-row result — the rule the engine’s JDBC driver applies — so `UPDATE` (which always answers a second “multi-joined” column) and `MERGE` (one column per action) report their true counts. `DDL` and `SELECT` report 0.
- Result columns map by declared type and actual values: integral `NUMBER` → integer, widening to double and then to the exact digit string when a value does not fit; scaled `NUMBER` and `FLOAT` → double; `BOOLEAN` → logical; `DATE` → Date; `TIMESTAMP*` → `POSIXct` labelled UTC (a trailing wire offset is honoured, so the instant never depends on the client machine’s timezone; a column with a fraction finer than a microsecond stays character); `TIME` → `difftime`; `BINARY` → a list of raw vectors; everything else, `VARIANT/OBJECT/ARRAY` included, arrives as character.

- `dbWriteTable` creates columns via `dbDataType` (override with `field.types`) and inserts multi-row VALUES batches; column names are quoted, so they round-trip through `dbReadTable` exactly.

Value

Each method returns what its **DBI** generic specifies:

- `dbConnect`: a `FrostlakeConnection`; `dbSendQuery` (and so `dbSendStatement`): a `FrostlakeResult`.
- `dbFetch`, `dbReadTable` and `dbColumnInfo`: a data frame. The column information has one row per result column, with name, type, precision, scale, length and nullable.
- `dbGetRowsAffected`: the number of rows the statement changed, 0 for a query or DDL; `dbGetRowCount`: the number of rows fetched so far; `dbGetStatement`: the statement text.
- `dbListTables` and `dbListFields`: a character vector; `dbDataType`: the engine type names as a character vector, named by column for a data frame.
- `dbIsValid`, `dbHasCompleted` and `dbExistsTable`: a single logical.
- `dbQuoteIdentifier`, `dbQuoteString` and `dbQuoteLiteral`: an SQL object.
- `dbGetInfo`: a named list — the driver and client versions for the driver; `db.version`, `dbname`, `username`, `host` and `port` for a connection.
- `dbAppendTable`: the number of rows written, invisibly; `dbBind`: the result, invisibly.
- `dbDisconnect`, `dbUnloadDriver`, `dbBegin`, `dbCommit`, `dbRollback`, `dbClearResult`, `dbWriteTable` and `dbRemoveTable`: TRUE, invisibly.
- `show`: prints a one-line description and returns NULL, invisibly.

See Also

[frostlake](#) for connecting.

Examples

```
# Type mapping needs no server
DBI::dbDataType(frostlake(), data.frame(flag = TRUE, n = 1L, x = 2.5, s = "x"))

## Not run:
# Everything below needs a running Frostlake DatabaseHttpServer
con <- DBI::dbConnect(frostlake(), url = "frostlake://localhost:18082/MY_DB")
res <- DBI::dbSendQuery(con, "SELECT 1 AS one")
DBI::dbColumnInfo(res)
DBI::dbFetch(res)
DBI::dbClearResult(res)
DBI::dbDisconnect(con)

## End(Not run)
```

Index

* methods

frostlake-methods, 3

dbAppendTable, FrostlakeConnection, character-method (frostlake-methods), 3

dbAppendTable, FrostlakeConnection, Id-method (frostlake-methods), 3

dbAppendTable, FrostlakeConnection, SQL-method (frostlake-methods), 3

dbBegin, FrostlakeConnection-method (frostlake-methods), 3

dbBind, FrostlakeResult-method (frostlake-methods), 3

dbClearResult, FrostlakeResult-method (frostlake-methods), 3

dbColumnInfo, FrostlakeResult-method (frostlake-methods), 3

dbCommit, FrostlakeConnection-method (frostlake-methods), 3

dbConnect, 2

dbConnect, FrostlakeDriver-method (frostlake-methods), 3

dbDataType, FrostlakeConnection-method (frostlake-methods), 3

dbDataType, FrostlakeDriver-method (frostlake-methods), 3

dbDisconnect, FrostlakeConnection-method (frostlake-methods), 3

dbExistsTable, FrostlakeConnection, character-method (frostlake-methods), 3

dbExistsTable, FrostlakeConnection, Id-method (frostlake-methods), 3

dbExistsTable, FrostlakeConnection, SQL-method (frostlake-methods), 3

dbFetch, FrostlakeResult-method (frostlake-methods), 3

dbGetInfo, FrostlakeConnection-method (frostlake-methods), 3

dbGetInfo, FrostlakeDriver-method (frostlake-methods), 3

dbGetRowCount, FrostlakeResult-method (frostlake-methods), 3

dbGetRowsAffected, FrostlakeResult-method (frostlake-methods), 3

dbGetStatement, FrostlakeResult-method (frostlake-methods), 3

dbHasCompleted, FrostlakeResult-method (frostlake-methods), 3

dbIsValid, FrostlakeConnection-method (frostlake-methods), 3

dbIsValid, FrostlakeDriver-method (frostlake-methods), 3

dbIsValid, FrostlakeResult-method (frostlake-methods), 3

dbListFields, FrostlakeConnection, character-method (frostlake-methods), 3

dbListFields, FrostlakeConnection, Id-method (frostlake-methods), 3

dbListFields, FrostlakeConnection, SQL-method (frostlake-methods), 3

dbListTables, FrostlakeConnection-method (frostlake-methods), 3

dbQuoteIdentifier, FrostlakeConnection, character-method (frostlake-methods), 3

dbQuoteIdentifier, FrostlakeConnection, Id-method (frostlake-methods), 3

dbQuoteIdentifier, FrostlakeConnection, SQL-method (frostlake-methods), 3

dbQuoteLiteral, FrostlakeConnection-method (frostlake-methods), 3

dbQuoteString, FrostlakeConnection, character-method (frostlake-methods), 3

dbQuoteString, FrostlakeConnection, SQL-method (frostlake-methods), 3

dbReadTable, FrostlakeConnection, character-method (frostlake-methods), 3

dbReadTable, FrostlakeConnection, Id-method (frostlake-methods), 3

dbReadTable, FrostlakeConnection, SQL-method

- (frostlake-methods), [3](#)
- dbRemoveTable,FrostlakeConnection,character-method
 - (frostlake-methods), [3](#)
- dbRemoveTable,FrostlakeConnection,Id-method
 - (frostlake-methods), [3](#)
- dbRemoveTable,FrostlakeConnection,SQL-method
 - (frostlake-methods), [3](#)
- dbRollback,FrostlakeConnection-method
 - (frostlake-methods), [3](#)
- dbSendQuery,FrostlakeConnection,character-method
 - (frostlake-methods), [3](#)
- dbUnloadDriver,FrostlakeDriver-method
 - (frostlake-methods), [3](#)
- dbWriteTable,FrostlakeConnection,character,data.frame-method
 - (frostlake-methods), [3](#)
- dbWriteTable,FrostlakeConnection,Id,data.frame-method
 - (frostlake-methods), [3](#)
- dbWriteTable,FrostlakeConnection,SQL,data.frame-method
 - (frostlake-methods), [3](#)
- frostlake, [2](#), [4](#)
- frostlake-methods, [3](#)
- frostlake-package (frostlake), [2](#)
- FrostlakeConnection-class (frostlake), [2](#)
- FrostlakeDriver-class (frostlake), [2](#)
- FrostlakeResult-class (frostlake), [2](#)
- show,FrostlakeConnection-method
 - (frostlake-methods), [3](#)
- show,FrostlakeDriver-method
 - (frostlake-methods), [3](#)
- show,FrostlakeResult-method
 - (frostlake-methods), [3](#)