Package ‘snowquery’

May 9, 2026

Title SQL Interface to 'Snowflake', 'Redshift', 'Postgres', 'SQLite', and 'DuckDB'

Version 1.3.0

Maintainer Dani Mermelstein <dmermelstein@hey.com>

Description Run 'SQL' queries across 'Snowflake', 'Amazon Redshift', 'PostgreSQL', 'SQLite', and 'DuckDB' from R with a single function. Optionally stream and cache large query results to a local 'DuckDB' database for efficient work with larger-than-memory datasets.

URL https://github.com/mermelstein/snowquery

BugReports https://github.com/mermelstein/snowquery/issues

Imports yaml, reticulate, RPostgres, RSQLite, DBI, duckdb, dplyr, dbplyr

Encoding UTF-8

RoxygenNote 7.3.3

License GPL (>= 3)

NeedsCompilation no

Author Dani Mermelstein [aut, cre, cph]

Repository CRAN

Date/Publication 2025-09-10 15:20:02 UTC

Contents

.cache_query_result *Cache a remote query result to a local DuckDB database*

Description

Efficiently streams the result of a query from a remote source (Snowflake, Redshift, Postgres) to a local DuckDB file. This method is memory-efficient and suitable for very large query results as it streams data without loading the entire result set into R's memory.

Usage

```
.cache_query_result(
  source_conn_name,
  source_query,
  dest_table_name,
  overwrite = TRUE,
  config_path = "~/snowquery_creds.yaml"
)
```

Arguments

source_conn_name	The name of the remote database connection in your snowquery_creds.yaml file.
source_query	The SQL query to execute on the remote source.
dest_table_name	The name of the table to be created in the local DuckDB database.
overwrite	A boolean (TRUE/FALSE) to control whether to overwrite the destination table if it already exists.
config_path	The path to your snowquery_creds.yaml file.

Value

Invisibly returns a confirmation message.

queryDB *Query a database*

Description

Run a SQL query on a Snowflake, Redshift or Postgres database and return the results as a data frame. See the snowquery README for more information on how to pass in your credentials.

Usage

```
queryDB(
  query,
  conn_name = "default",
  db_type = NULL,
  username = NULL,
  password = NULL,
  host = NULL,
  port = NULL,
  database = NULL,
  warehouse = NULL,
  account = NULL,
  role = NULL,
  sslmode = NULL,
  timeout = 15,
  cache_table_name = NULL,
  overwrite = TRUE
)
```

Arguments

query	A string of the SQL query to execute
conn_name	The name of the connection to use in snowquery_creds.yaml (e.g. "my_snowflake_dwh")
db_type	The type of database to connect to (e.g. "snowflake", "redshift" or "postgres")
username	The username to use for authentication
password	The password to use for authentication
host	The hostname or IP address of the database server
port	The port number to use for the database connection
database	The name of the database to connect to
warehouse	Snowflake The name of the warehouse to use for the Snowflake connection
account	Snowflake The name of the Snowflake account to connect to
role	Snowflake The name of the role to use for the Snowflake connection
sslmode	Whether to use sslmode for the postgres or redshift connection
timeout	The number of seconds to wait for the database to connect successfully
cache_table_name	The name of the table to create inside the DuckDB file. If provided, the query result is streamed directly to DuckDB and a confirmation message is returned instead of a data frame.
overwrite	A boolean (TRUE/FALSE) to control whether to overwrite an existing table in the cache.

Value

A data frame containing the results of the query, or a confirmation message if `cache_table_name` is used.

Examples

```
## Not run:
# Query the database and get a dataframe of results
result <- queryDB("SELECT * FROM my_table", conn_name='my_snowflake_dwh')
print(result)

## End(Not run)
## Not run:
# Stream a large query result directly to the local DuckDB cache
queryDB("SELECT * FROM very_large_table",
        conn_name = 'my_snowflake_dwh',
        cache_table_name = 'large_table_local',
        overwrite = TRUE)

## End(Not run)
```

Index