Package: polarssql (via r-universe)

March 20, 2025

Title A 'polars' backend for 'DBI'.

Description DBI-compliant interface to 'polars'.

Version 0.0.0.9000

License MIT + file LICENSE

URL https://rpolars.github.io/r-polarssql/, https://github.com/rpolars/r-polarssql

BugReports https://github.com/rpolars/r-polarssql/issues

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Depends R (>= 4.2)

Imports DBI (>= 1.2.1), methods, polars (>= 0.13.0), rlang

Suggests dplyr, dbplyr, nanoarrow, patrick, testthat (>= 3.0.0), DBItest (>= 1.7.3)

Config/testthat/edition 3

Collate 'polarssql-package.R' 'Driver.R' 'Connection.R' 'Result.R' 'dbClearResult_Result.R' 'dbConnect_Driver.R' 'dbDisconnect_Connectsion.R' 'dbExistsTable_Connection_character.R' 'dbFetch_Result.R' 'dbGetRowsAffected_Result.R' 'dbHasCompleted_Result.R' 'dbIsValid_Connection.R' 'dbIsValid_Driver.R' 'dbIsValid_Result.R' 'dbListFields_Connection_character.R' 'dbListTables_Connection.R' 'dbQuoteIdentifier_Connection_character.R' 'dbQuoteString_Connection_character.R' 'dbRemoveTable_Connection_character.R' 'dbSendQuery_Connection_character.R' 'dbWriteTable_Connection_character.R' 'default_connection.R' 'pkg-dbplyr.R' 'query.R' 'register.R' 'show_Connection.R' 'show_Driver.R' 'show_Result.R' 'utils.R' 'zzz.R'

Repository https://rpolars.r-universe.dev

RemoteUrl https://github.com/rpolars/r-polarssql

RemoteRef HEAD

RemoteSha 4aebf49ed8a31191c902027747879924a2c40b06

Contents

`as_polars_lf.tbl_polarssql_connection`
Compute results of a dbplyr query

Description

Compute results of a dbplyr query

Usage

```
## S3 method for class 'tbl_polarssql_connection'
as_polars_lf(x, ..., cte = TRUE)

## S3 method for class 'tbl_polarssql_connection'
as_polars_df(x, ..., cte = TRUE)

## S3 method for class 'tbl_polarssql_connection'
compute(x, ..., n = Inf, cte = TRUE)

## S3 method for class 'tbl_polarssql_connection'
as.data.frame(x, ..., cte = TRUE)
```

Arguments

`x`	A tbl_polarssql_connection object.
`...`	• For `as_polars_lf(<tbl_polarssql_connection>)`: Ignored. • For Other functions: Other arguments passed to `as_polars_df(<RPolarsLazyFrame>)`.
`cte`	**[Experimental]** Use common table expressions in the generated SQL?
`n`	Number of rows to fetch. Defaults to `Inf`, meaning all rows.

Examples

```
library(dplyr, warn.conflicts = FALSE)
library(polars)

t <- tbl_polarssql(mtcars) |>
  filter(cyl == 4)

as_polars_lf(t)

as_polars_df(t, n_rows = 1)

compute(t, n = 1) # Equivalent to `as_polars_df(t, n_rows = 1)`

as.data.frame(t, n_rows = 1)

# Clean up
DBI::dbDisconnect(polarssql_default_connection())
```

DBI *DBI methods*

Description

Implementations of pure virtual functions defined in the DBI package.

Usage

```
## S4 method for signature 'polarssql_result'
dbClearResult(res, ...)

## S4 method for signature 'polarssql_driver'
dbConnect(drv, ...)

## S4 method for signature 'polarssql_connection'
dbDisconnect(conn, ...)

## S4 method for signature 'polarssql_connection,character'
dbExistsTable(conn, name, ...)

## S4 method for signature 'polarssql_result'
dbFetch(res, n = -1, ...)

## S4 method for signature 'polarssql_result'
dbGetRowsAffected(res, ...)

## S4 method for signature 'polarssql_result'
dbHasCompleted(res, ...)
```

```
## S4 method for signature 'polarssql_connection'
dbIsValid(dbObj, ...)

## S4 method for signature 'polarssql_driver'
dbIsValid(dbObj, ...)

## S4 method for signature 'polarssql_result'
dbIsValid(dbObj, ...)

## S4 method for signature 'polarssql_connection,character'
dbListFields(conn, name, ...)

## S4 method for signature 'polarssql_connection'
dbListTables(conn, ...)

## S4 method for signature 'polarssql_connection,character'
dbQuoteIdentifier(conn, x, ...)

## S4 method for signature 'polarssql_connection,character'
dbQuoteString(conn, x, ...)

## S4 method for signature 'polarssql_connection,character'
dbRemoveTable(conn, name, ..., fail_if_missing = TRUE)

## S4 method for signature 'polarssql_connection,character'
dbSendQuery(conn, statement, ...)

## S4 method for signature 'polarssql_connection,character,data.frame'
dbWriteTable(conn, name, value, ..., overwrite = FALSE)

## S4 method for signature 'polarssql_connection'
show(object)

## S4 method for signature 'polarssql_driver'
show(object)

## S4 method for signature 'polarssql_result'
show(object)
```

Arguments

res	An object inheriting from DBIResult.
...	Other arguments passed on to methods.
drv	an object that inherits from DBIDriver, or an existing DBIConnection object (in order to clone an existing connection).
conn	A DBIConnection object, as returned by dbConnect().
name	The table name, passed on to dbQuoteIdentifier(). Options are:

- a character string with the unquoted DBMS table name, e.g. `"table_name"`,
- a call to `Id()` with components to the fully qualified table name, e.g. `Id(schema = "my_schema", table = "table_name")`
- a call to `SQL()` with the quoted and fully qualified table name given verbatim, e.g. `SQL('"my_schema"."table_name"')`

`n`	maximum number of records to retrieve per fetch. Use `n = -1` or `n = Inf` to retrieve all pending records. Some implementations may recognize other special values.
`dbObj`	An object inheriting from DBIObject, i.e. DBIDriver, DBIConnection, or a DBIResult
`x`	A character vector, SQL or Id object to quote as identifier.
`fail_if_missing`	If `FALSE`, `dbRemoveTable()` succeeds if the table doesn't exist.
`statement`	a character string containing SQL.
`value`	A data.frame (or coercible to data.frame).
`overwrite`	Allow overwriting the destination table. Cannot be `TRUE` if append is also `TRUE`.
`object`	Any R object

`dbplyr-backend-polarssql` *polarssql backend for dbplyr*

Description

Use `simulate_polarssql()` with `dbplyr::tbl_lazy()` or `dbplyr::lazy_frame()` to see simulated SQL without converting to live access. `tbl_polarssql()` is similar to `dbplyr::tbl_memdb()`, but the backend is Polars instead of SQLite. It uses `polarssql_default_connection()` as the DBI connection.

Usage

```
simulate_polarssql()

tbl_polarssql(df, name = deparse(substitute(df)), ..., overwrite = FALSE)
```

Arguments

`df`	Data frame to copy
`name`	Name of table in database: defaults to a random name that's unlikely to conflict with an existing table.
`...`	Ignored.
`overwrite`	If `TRUE`, overwrite the existing table which has the same name. If not `TRUE` (default), skip writing a table if it already exists.

Examples

```
library(dplyr, warn.conflicts = FALSE)

# Test connection shows the SQL query.
dbplyr::tbl_lazy(mtcars, simulate_polarssql(), name = "mtcars") |>
  filter(cyl == 4) |>
  arrange(desc(mpg)) |>
  select(contains("c")) |>
  head(n = 3)

# Actual polarssql connection shows the Polars naive plan (LazyFrame).
tbl_polarssql(mtcars) |>
  filter(cyl == 4) |>
  arrange(desc(mpg)) |>
  select(contains("c")) |>
  head(n = 3)

# Unlike other dbplyr backends, `compute` has a special behavior.
# It returns a polars DataFrame.
tbl_polarssql(mtcars) |>
  filter(cyl == 4) |>
  arrange(desc(mpg)) |>
  select(contains("c")) |>
  head(n = 3) |>
  compute()
```

`polarssql` *polarssql driver*

Description

`polarssql()` creates a DBI driver instance.

Usage

```
polarssql()
```

Examples

```
polarssql()
```

polarssql_default_connection *Get the default connection*

Description

Get the default built-in connection.

Usage

```
polarssql_default_connection()
```

Value

A polarssql connection object

Examples

```
# Clean up
DBI::dbDisconnect(polarssql_default_connection())

polarssql_default_connection()

# Register a Table
polarssql_register(mtcars = mtcars)

polarssql_default_connection()

# Clean up
polarssql_unregister("mtcars")

polarssql_default_connection()
```

polarssql_query *Execute SQL query*

Description

Execute SQL query

Usage

```
polarssql_query(sql, conn = polarssql_default_connection())
```

Arguments

sql	A SQL string.
conn	A polarssql connection, created by polarssql(). Use the default built-in connection by default.

Value

polars LazyFrame

Examples

```
polarssql_register(mtcars = mtcars)

query <- "SELECT * FROM mtcars LIMIT 5"

# Returns a polars LazyFrame
polarssql_query(query)

# Clean up
polarssql_unregister("mtcars")
```

---

polarssql_register *Register data frames as tables*

Description

Register data frames as tables

Usage

```
polarssql_register(
  ...,
  .conn = polarssql_default_connection(),
  .overwrite = FALSE
)

polarssql_unregister(names, conn = polarssql_default_connection())
```

Arguments

...	<dynamic-dots> Name-value pairs of data.frame like objects to register.
.conn, conn	A polarssql connection, created by polarssql(). Use the default built-in connection by default.
.overwrite	Should an existing registration be overwritten?
names	Names of the tables to unregister.

Value

The polarssql connection invisibly.

Examples

```
con <- dbConnect(polarssql())

polarssql_register(df1 = mtcars, df2 = mtcars, .conn = con)
con

polarssql_unregister(c("df1", "df2"), conn = con)
con
```