Package 'duckdb'

March 16, 2021

Title DBI Package for the DuckDB Database Management System

Version 0.2.5

Description The DuckDB project is an embedded analytical data
management system with support for the Structured Query Language (SQL). This package includes all of
DuckDB and a R Database Interface (DBI) connector.

License MPL

URL https://duckdb.org/, https://github.com/cwida/duckdb

BugReports https://github.com/cwida/duckdb/issues

Depends DBI, R (>= 3.5.0)

Imports methods, utils

Suggests callr, DBItest, dbplyr, nycflights13, testthat, withr

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

SystemRequirements C++11, GCC on Solaris

NeedsCompilation yes

Author Hannes Mühleisen [aut, cre] (<https://orcid.org/0000-0001-8552-0029>),
Mark Raasveldt [aut] (<https://orcid.org/0000-0001-5005-6844>),
DuckDB Contributors [aut],
Apache Software Foundation [cph],
PostgreSQL Global Development Group [cph],
The Regents of the University of California [cph],
Cameron Desrochers [cph],
Victor Zverovich [cph],
RAD Game Tools [cph],
Valve Software [cph],
Rich Geldreich [cph],
Tenacious Software LLC [cph],
The RE2 Authors [cph],
Google Inc. [cph],

Facebook Inc. [cph],
Steven G. Johnson [cph],
Jiahao Chen [cph],
Tony Kelman [cph],
Jonas Fonseca [cph],
Lukas Fittl [cph]

Maintainer Hannes Mühleisen <hannes@cwi.nl>

Repository CRAN

Date/Publication 2021-03-16 15:10:02 UTC

R topics documented:

`duckdb-package` *DuckDB client package for R*

Description

R client package for DuckDB: an embeddable SQL OLAP Database Management System.

See Also

`duckdb()` for connection instructions.

`https://duckdb.org/` for the project website.

`dbConnect,duckdb_driver-method`
Connect to a DuckDB database instance

Description

`dbConnect()` connects to a database instance.

`dbDisconnect()` closes a DuckDB database connection, optionally shutting down the associated instance.

`duckdb()` creates or reuses a database instance.

`duckdb_shutdown()` shuts down a database instance.

Usage

```
## S4 method for signature 'duckdb_driver'
dbConnect(
  drv,
  dbdir = DBDIR_MEMORY,
  ...,
  debug = getOption("duckdb.debug", FALSE),
  read_only = FALSE,
  timezone_out = "UTC",
  tz_out_convert = c("with", "force")
)

## S4 method for signature 'duckdb_connection'
dbDisconnect(conn, ..., shutdown = FALSE)

duckdb(dbdir = DBDIR_MEMORY, read_only = FALSE)

duckdb_shutdown(drv)
```

Arguments

drv	Object returned by duckdb()
dbdir	Location for database files. Should be a path to an existing directory in the file system. With the default, all data is kept in RAM
...	Ignored
debug	Print additional debug information such as queries
read_only	Set to TRUE for read-only operation
timezone_out	The time zone returned to R, defaults to "UTC", which is currently the only timezone supported by duckdb. If you want to display datetime values in the local timezone, set to Sys.timezone() or "".
tz_out_convert	How to convert timestamp columns to the timezone specified in timezone_out. There are two options: "with", and "force". If "with" is chosen, the timestamp will be returned as it would appear in the specified time zone. If "force" is chosen, the timestamp will have the same clock time as the timestamp in the database, but with the new time zone.
conn	A duckdb_connection object
shutdown	Set to TRUE to shut down the DuckDB database instance that this connection refers to.

Value

dbConnect() returns an object of class duckdb_connection.

duckdb() returns an object of class duckdb_driver.

dbDisconnect() and duckdb_shutdown() are called for their side effect.

Examples

```
drv <- duckdb()
con <- dbConnect(drv)

dbGetQuery(con, "SELECT 'Hello, world!'")

dbDisconnect(con)
duckdb_shutdown(drv)

# Shorter:
con <- dbConnect(duckdb())
dbGetQuery(con, "SELECT 'Hello, world!'")
dbDisconnect(con, shutdown = TRUE)
```

---

duckdb_read_csv *Reads a CSV file into DuckDB*

Description

Directly reads a CSV file into DuckDB, tries to detect and create the correct schema for it. This usually is much faster than reading the data into R and writing it to DuckDB.

Usage

```
duckdb_read_csv(
  conn,
  name,
  files,
  header = TRUE,
  na.strings = "",
  nrow.check = 500,
  delim = ",",
  quote = "\"",
  col.names = NULL,
  lower.case.names = FALSE,
  sep = delim,
  transaction = TRUE,
  ...
)
```

Arguments

conn	A DuckDB connection, created by dbConnect().
name	The name for the virtual table that is registered or unregistered
files	One or more CSV file names, should all have the same structure though
header	Whether or not the CSV files have a separate header in the first line

na.strings	Which strings in the CSV files should be considered to be NULL
nrow.check	How many rows should be read from the CSV file to figure out data types
delim	Which field separator should be used
quote	Which quote character is used for columns in the CSV file
col.names	Override the detected or generated column names
lower.case.names	Transform column names to lower case
sep	Alias for delim for compatibility
transaction	Should a transaction be used for the entire operation
...	Passed on to read.csv()

Value

The number of rows in the resulted table, invisibly.

Examples

```
con <- dbConnect(duckdb())

data <- data.frame(a = 1:3, b = letters[1:3])
path <- tempfile(fileext = ".csv")

write.csv(data, path, row.names = FALSE)

duckdb_read_csv(con, "data", path)
dbReadTable(con, "data")

dbDisconnect(con)
```

---

duckdb_register *Register a data frame as a virtual table*

Description

duckdb_register() registers a data frame as a virtual table (view) in a DuckDB connection. No data is copied.

Usage

```
duckdb_register(conn, name, df)

duckdb_unregister(conn, name)
```

Arguments

conn	A DuckDB connection, created by dbConnect().
name	The name for the virtual table that is registered or unregistered
df	A data.frame with the data for the virtual table

Details

duckdb_unregister() unregisters a previously registered data frame.

Value

These functions are called for their side effect.

Examples

```
con <- dbConnect(duckdb())

data <- data.frame(a = 1:3, b = letters[1:3])

duckdb_register(con, "data", data)
dbReadTable(con, "data")

duckdb_unregister(con, "data")
try(dbReadTable(con, "data"))

dbDisconnect(con)
```

Index