

Package ‘duckdb’

September 6, 2021

Title DBI Package for the DuckDB Database Management System

Version 0.2.9

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/duckdb/duckdb>

BugReports <https://github.com/duckdb/duckdb/issues>

Depends DBI, R (>= 3.6.0)

Imports methods, utils

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

Encoding UTF-8

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],
 Salvatore Sanfilippo [cph],
 Art.sy, Inc. [cph],
 Oran Agra [cph],
 Redis Labs, Inc. [cph],
 Melissa O'Neill [cph],
 PCG Project contributors [cph]

Maintainer Hannes Mühleisen <hannes@cwil.nl>

Repository CRAN

Date/Publication 2021-09-06 08:40:08 UTC

R topics documented:

duckdb-package	2
dbConnect,duckdb_driver-method	3
duckdb_read_csv	4
duckdb_register	6
duckdb_register_arrow	7

Index	8
--------------	----------

duckdb-package	<i>DuckDB client package for R</i>
----------------	------------------------------------

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"),
  config = list()
)
```

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

```
duckdb(dbdir = DBDIR_MEMORY, read_only = FALSE, config = list())
```

```
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 <code>Sys.timezone()</code> or "".

tz_out_convert	How to convert timestamp columns to the timezone specified in <code>timezone_out</code> . 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.
config	Named list with DuckDB configuration flags
conn	A <code>duckdb_connection</code> 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)
```

<code>duckdb_read_csv</code>	<i>Reads a CSV file into DuckDB</i>
------------------------------	-------------------------------------

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

<code>conn</code>	A DuckDB connection, created by <code>dbConnect()</code> .
<code>name</code>	The name for the virtual table that is registered or unregistered
<code>files</code>	One or more CSV file names, should all have the same structure though
<code>header</code>	Whether or not the CSV files have a separate header in the first line
<code>na.strings</code>	Which strings in the CSV files should be considered to be NULL
<code>nrow.check</code>	How many rows should be read from the CSV file to figure out data types
<code>delim</code>	Which field separator should be used
<code>quote</code>	Which quote character is used for columns in the CSV file
<code>col.names</code>	Override the detected or generated column names
<code>lower.case.names</code>	Transform column names to lower case
<code>sep</code>	Alias for <code>delim</code> for compatibility
<code>transaction</code>	Should a transaction be used for the entire operation
<code>...</code>	Passed on to <code>read.csv()</code>

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	<i>Register a data frame as a virtual table</i>
-----------------	---

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)
```

duckdb_register_arrow *Register an Arrow data source as a virtual table*

Description

duckdb_register_arrow() registers an Arrow data source as a virtual table (view) in a DuckDB connection. No data is copied.

Usage

```
duckdb_register_arrow(conn, name, arrow_scannable)
```

```
duckdb_unregister_arrow(conn, name)
```

Arguments

conn	A DuckDB connection, created by dbConnect().
name	The name for the virtual table that is registered or unregistered
arrow_scannable	A scannable Arrow-object

Details

duckdb_unregister_arrow() unregisters a previously registered data frame.

Value

These functions are called for their side effect.

Index

`dbConnect`, `duckdb_driver`-method, [3](#)
`dbDisconnect`, `duckdb_connection`-method
 (`dbConnect`, `duckdb_driver`-method),
 [3](#)
`duckdb`
 (`dbConnect`, `duckdb_driver`-method),
 [3](#)
`duckdb()`, [2](#)
`duckdb`-package, [2](#)
`duckdb_connection`, [4](#)
`duckdb_driver`, [4](#)
`duckdb_read_csv`, [4](#)
`duckdb_register`, [6](#)
`duckdb_register_arrow`, [7](#)
`duckdb_shutdown`
 (`dbConnect`, `duckdb_driver`-method),
 [3](#)
`duckdb_unregister` (`duckdb_register`), [6](#)
`duckdb_unregister_arrow`
 (`duckdb_register_arrow`), [7](#)

`read.csv()`, [5](#)

`Sys.timezone()`, [3](#)