## Package ‘RPresto’

November 1, 2023

**Title**  DBI Connector to Presto

**Version**  1.4.6

**Copyright**  Meta Platforms, Inc. 2015-present.

**Description**  Implements a 'DBI' compliant interface to Presto. Presto is an open source distributed SQL query engine for running interactive analytic queries against data sources of all sizes ranging from gigabytes to petabytes: <https://prestodb.io/>.

**Depends**  R (>= 3.1.0), methods

**Imports**  DBI (>= 0.3.0), httr (>= 0.6), openssl, jsonlite, stringi, stats, utils, purrr, dplyr (>= 0.7.0), dbplyr (>= 2.3.3), tibble, bit64, rlang, lifecycle, lubridate, progress, vctrs

**Suggests**  testthat, hms, knitr, rmarkdown

**License**  BSD_3_clause + file LICENSE

**URL**  https://github.com/prestodb/RPresto

**BugReports**  https://github.com/prestodb/RPresto/issues

**Encoding**  UTF-8

**Collate**  'PrestoDriver.R' 'Presto.R' 'PrestoSession.R'
  'PrestoConnection.R' 'PrestoQuery.R' 'PrestoResult.R'
  'RPresto-package.R' 'chunk.R' 'create.dummy.tables.R' 'cte.R'
  'dbAppendTable.R' 'dbClearResult.R' 'dbConnect.R'
  'dbCreateTable.R' 'dbCreateTableAs.R' 'dbDataType.R'
  'dbDisconnect.R' 'dbExistsTable.R' 'dbFetch.R' 'dbGetInfo.R'
  'dbGetQuery.R' 'dbGetRowCount.R' 'dbGetRowsAffected.R'
  'dbGetStatement.R' 'dbHasCompleted.R' 'dbIsValid.R'
  'dbListFields.R' 'dbListTables.R' 'dbQuoteIdentifier.R'
  'dbQuoteLiteral.R' 'dbReadTable.R' 'dbRemoveTable.R'
  'dbRenameTable.R' 'dbSendQuery.R' 'dbUnloadDriver.R'
  'dbWriteTable.R' 'dbplyr-db.R' 'dbplyr-sql.R' 'dbplyr-src.R'
  'default.R' 'fetch.R' 'presto.field.R'
  'presto.field_utilities.R' 'request_headers.R'
  'sqlCreateTable.R' 'sqlCreateTableAs.R' 'zzz.R'

**RoxygenNote**  7.2.3
**add_chunk**

Add a chunk field to a data frame

### Description

This auxiliary function adds a field, if necessary, to a data frame so that each compartment of the data frame that corresponds to a unique combination of the chunk fields has a size below a certain threshold. This resulting data frame can then be safely used in `dbAppendTable()` because Presto has a size limit on any discrete INSERT INTO statement.

### Usage

```r
add_chunk(
  value,
  base_chunk_fields = NULL,
  chunk_size = 1e+06,
  new_chunk_field_name = "aux_chunk_idx"
)
```
Arguments

value  The original data frame.
base_chunk_fields  A character vector of existing field names that are used to split the data frame before checking the chunk size.
chunk_size  Maximum size (in bytes) of the VALUES statement encoding each unique chunk. Default to 1,000,000 bytes (i.e. 1Mb).
new_chunk_field_name  A string indicating the new chunk field name. Default to "aux_chunk_idx".

Examples

## Not run:
# returns the original data frame because it's within size
add_chunk(iris)
# add a new aux_chunk_idx field
add_chunk(iris, chunk_size = 2000)
# the new aux_chunk_idx field is added on top of Species
add_chunk(iris, chunk_size = 2000, base_chunk_fields = c("Species"))

## End(Not run)

---

**dbCreateTableAs**  
Create a table in database using a statement

Description

Create a table in database using a statement

Usage

dbCreateTableAs(conn, name, sql, overwrite = FALSE, with = NULL, ...)

Arguments

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""
sql  a character string containing SQL statement.
overwrite  A boolean indicating if an existing table should be overwritten. Default to FALSE.
with  An optional WITH clause for the CREATE TABLE statement.
...  Other parameters passed on to methods.
dbDataType,PrestoDriver-method

*Return the corresponding presto data type for the given R object*

Description

Return the corresponding presto data type for the given R object

Usage

```r
## S4 method for signature 'PrestoDriver'
dbDataType(dbObj, obj, ...)
```

Arguments

- `dbObj` A `PrestoDriver` object
- `obj` Any R object
- `...` Extra optional parameters, not currently used

Details

The default value for unknown classes is `VARCHAR`.

Value

A character value corresponding to the Presto type for `obj`

Examples

```r
drv <- RPresto::Presto()
dbDataType(drv, 1)
dbDataType(drv, NULL)
dbDataType(drv, as.POSIXct("2015-03-01 00:00:00", tz = "UTC"))
dbDataType(drv, Sys.time())
dbDataType(drv, list(
  c("a" = 1L, "b" = 2L),
  c("a" = 3L, "b" = 4L)
))
dbDataType(drv, list(
  c(as.Date("2015-03-01"), as.Date("2015-03-02")),
  c(as.Date("2016-03-01"), as.Date("2016-03-02"))
))
dbDataType(drv, iris)
```
**Description**

Metadata about database objects

For the **PrestoResult** object, the implementation returns the additional **stats** field which can be used to implement things like progress bars. See the examples section.

**Usage**

```r
## S4 method for signature 'PrestoDriver'
dbGetInfo(dbObj)

## S4 method for signature 'PrestoConnection'
dbGetInfo(dbObj)

## S4 method for signature 'PrestoResult'
dbGetInfo(dbObj)
```

**Arguments**

- **dbObj** A **PrestoDriver**, **PrestoConnection** or **PrestoResult** object

**Value**

- **PrestoResult** A **list()** with elements
  - **statement** The SQL sent to the database
  - **row.count** Number of rows fetched so far
  - **has.completed** Whether all data has been fetched
  - **stats** Current stats on the query

**Examples**

```r
## Not run:
conn <- dbConnect(Presto(), "localhost", 7777, "onur", "datascience")
result <- dbSendQuery(conn, "SELECT * FROM jonchang_iris")
iris <- data.frame()
progress.bar <- NULL
while (!dbHasCompleted(result)) {
  chunk <- dbFetch(result)
  if (!NROW(iris)) {
    iris <- chunk
  } else if (NROW(chunk)) {
    iris <- rbind(iris, chunk)
  } else {
    iris <- iris
  }
  progress.bar <- 100 * NROW(iris) / NROW(result)
  cat("Progress:", progress.bar, "% completed.
```

stats <- dbGetInfo(result)["stats"]
if (!is.null(progress.bar)) {
  progress.bar <- txtProgressBar(0, stats["totalSplits"], style = 3)
} else {
  setTxtProgressBar(progress.bar, stats["completedSplits"])
}
}
close(progress.bar)

## End(Not run)

dbpalyr_edition.PrestoConnection

*Inform the dbplyr version used in this package*

**Description**
Inform the dbplyr version used in this package

**Usage**

```
## S3 method for class 'PrestoConnection'
dbpalyr_edition(con)
```

**Arguments**

- **con**
  A DBIConnection object.

---

dbRenameTable

*Rename a table*

**Description**
Rename a table

**Usage**

```
dbRenameTable(conn, name, new_name, ...)
```

**Arguments**

- **conn**
  A PrestoConnection.
- **name**
  Existing table's name.
- **new_name**
  New table name.
- **...**
  Extra arguments passed to dbExecute.
db_list_tables.PrestoConnection

**dbplyr database methods**

### Description

dplyr database methods

### Usage

```r
## S3 method for class 'PrestoConnection'
db_list_tables(con)

## S3 method for class 'PrestoConnection'
db_has_table(con, table)

## S3 method for class 'PrestoConnection'
db_write_table(
  con,
  table,
  types,
  values,
  temporary = FALSE,
  overwrite = FALSE,
  ...
)

## S3 method for class 'PrestoConnection'
db_copy_to(
  con,
  table,
  values,
  overwrite = FALSE,
  types = NULL,
  temporary = TRUE,
  unique_indexes = NULL,
  indexes = NULL,
  analyze = TRUE,
  ...
)

## S3 method for class 'PrestoConnection'
db_compute(
  con,
```
kerberos_configs

A convenient wrapper around Kerberos config

Description

The configs specify authentication protocol and additional settings.

Usage

kerberos_configs(user = "", password = "", service_name = "presto")

Arguments

user User name to pass to http::authenticate(). Default to "".
password Password to pass to http::authenticate(). Default to "".
service_name The service name. Default to "presto".

---

## S3 method for class 'PrestoConnection'
db_sql_render(con, sql, ..., use_presto_cte = TRUE)

Arguments

con A PrestoConnection as returned by dbConnect().
table Table name
types Column types. If not provided, column types are inferred using dbDataType.
values A data.frame.
temporary If a temporary table should be used. Not supported. Only FALSE is accepted.
overwrite If an existing table should be overwritten.
... Extra arguments to be passed to individual methods.
with An optional WITH clause for the CREATE TABLE statement.
unique_indexes, indexes, analyze, in_transaction Ignored. Included for compatibility with generics.
sql A SQL statement.
use_presto_cte [Experimental] A logical value indicating if to use common table expressions stored in PrestoConnection when possible. Default to TRUE. See vignette("common-table-expressions")
Value

A `httr::config()` output that can be passed to the request.config argument of `dbConnect()`.

---

**Presto**  
*Connect to a Presto database*

Description

Connect to a Presto database

Usage

```
Presto(...)
```

```r
## S4 method for signature 'PrestoDriver'
dbConnect(
  drv,
  catalog,
  schema,
  user,
  host = "localhost",
  port = 8080,
  source = methods::packageName(),
  session.timezone = "",
  output.timezone = "",
  parameters = list(),
  ctes = list(),
  request.config = httr::config(),
  use.trino.headers = FALSE,
  extra.credentials = "",
  bigint = c("integer", "integer64", "numeric", "character"),
  ...
)
```

```
## S4 method for signature 'PrestoConnection'
dbDisconnect(conn)
```

Arguments

... currently ignored
drv A driver object generated by `Presto()`
catalog The catalog to be used
schema The schema to be used
user The current user
host The presto host to connect to
port Port to use for the connection
source Source to specify for the connection
session.timezone Time zone of the Presto server. Presto returns timestamps without time zones with respect to this value. The time arithmetic (e.g. adding hours) will also be done in the given time zone. This value is passed to Presto server via the request headers.
output.timezone The time zone using which TIME WITH TZ and TIMESTAMP values in the output should be represented. Default to the Presto server timezone (use `show(<PrestoConnection>)` to see).
parameters A list() of extra parameters to be passed in the ‘X-Presto-Session’ header
tces [Experimental] A list of common table expressions (CTEs) that can be used in the WITH clause. See vignette("common-table-expressions").
request.config An optional config list, as returned by `httr::config()`, to be sent with every HTTP request.
use.trino.headers A boolean to indicate whether Trino request headers should be used. Default to FALSE.
extra.credentials Extra credentials to be passed in the X-Presto-Extra-Credential or X-Trino-Extra-Credential header (depending on the value of the use.trino.headers argument). Default to an empty string.
bigint The R type that Presto’s 64-bit integer (BIGINT) class should be translated to. The default is "integer", which returns R’s integer type, but results in NA for values above/below +/-2147483647. "integer64" returns a `bit64::integer64`, which allows the full range of 64 bit integers. "numeric" coerces into R’s double type but might result in precision loss. Lastly, "character" casts into R’s character type.
conn A PrestoConnection object

Value

Presto A PrestoDriver object
dbConnect A PrestoConnection object
dbDisconnect A logical() value indicating success

Examples

```r
# Not run:
conn <- dbConnect(Presto(),
  catalog = "hive", schema = "default",
  user = "onur", host = "localhost", port = 8080,
  session.timezone = "US/Eastern", bigint = "character"
)
dbListTables(conn, "%_iris")
dbDisconnect(conn)
```
presto_default

Check if default database is available.

Description

presto_default() works similarly but returns a connection on success and throws a testthat skip condition on failure, making it suitable for use in tests.

R Presto examples and tests connect to a default database via dbConnect(Presto(), ...). This function checks if that database is available, and if not, displays an informative message.

Usage

presto_default(...)

presto_has_default(...)

Arguments

... Additional arguments passed on to dbConnect()

Examples

if (presto_has_default()) {
  db <- presto_default()
  print(dbListTables(db))
  dbDisconnect(db)
} else {
  message("No database connection."
}

sqlCreateTableAs

Compose query to create a simple table using a statement

Description

Compose query to create a simple table using a statement

Usage

sqlCreateTableAs(con, name, sql, with = NULL, ...)

Arguments

con  A database connection.
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")`

sql A character string containing SQL statement.
with An optional WITH clause for the CREATE TABLE statement.

... Other arguments used by individual methods.

sql_query_save.PrestoConnection
dplyr SQL methods

Description
dplyr SQL methods

Usage
```r
## S3 method for class 'PrestoConnection'
sql_query_save(con, sql, name, temporary = TRUE, ..., with = NULL)
```

Arguments

con   A database connection.
sql   a character string containing SQL statement.
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")`
temporary If a temporary table should be created. Default to TRUE in the `dplyr::sql_query_save()` generic. The default value generates an error in Presto. Using `temporary = FALSE` to save the query in a permanent table.
... Other arguments used by individual methods.
with An optional WITH clause for the CREATE TABLE statement.
Description

Allows you to connect to an existing database through a presto connection.

Usage

```r
src_presto(
  catalog = NULL,
  schema = NULL,
  user = NULL,
  host = NULL,
  port = NULL,
  source = NULL,
  session.timezone = NULL,
  parameters = NULL,
  bigint = c("integer", "integer64", "numeric", "character"),
  con = NULL,
  ...
)
```

Arguments

catalog Catalog to use in the connection
schema Schema to use in the connection
user User name to use in the connection
host Host name to connect to the database
port Port number to use with the host name
source Source to specify for the connection
session.timezone Time zone for the connection
parameters Additional parameters to pass to the connection
bigint The R type that Presto’s 64-bit integer (BIGINT) types should be translated to. The default is "integer", which returns R’s integer type, but results in NA for values above/below +/-2147483647. "integer64" returns a `bit64::integer64`, which allows the full range of 64 bit integers. "numeric" coerces into R’s double type but might result in precision loss. Lastly, "character" casts into R’s character type.
con An object that inherits from `PrestoConnection`, typically generated by `DBI::dbConnect`. When a valid connection object is supplied, Other arguments are ignored.
... For `src_presto` other arguments passed on to the underlying database connector `dbConnect`. For `tbl.src_presto`, it is included for compatibility with the generic, but otherwise ignored.
Examples

## Not run:
# To connect to a database
my_db <- src_presto(
    catalog = "memory",
    schema = "default",
    user = Sys.getenv("USER"),
    host = "http://localhost",
    port = 8080,
    session.timezone = "Asia/Kathmandu"
)
# Use a PrestoConnection
my_con <- DBI::dbConnect(
    catalog = "memory",
    schema = "default",
    user = Sys.getenv("USER"),
    host = "http://localhost",
    port = 8080,
    session.timezone = "Asia/Kathmandu"
)
my_db2 <- src_presto(con = my_con)

## End(Not run)
Index

add_chunk, 2
bit64::integer64, 10, 13
db_compute.PrestoConnection
   (db_list_tables.PrestoConnection), 7
db_copy_to.PrestoConnection
   (db_list_tables.PrestoConnection), 7
db_has_table.PrestoConnection
   (db_list_tables.PrestoConnection), 7
db_list_tables.PrestoConnection, 7
db_sql_render.PrestoConnection
   (db_list_tables.PrestoConnection), 7
db_write_table.PrestoConnection
   (db_list_tables.PrestoConnection), 7
dbConnect, 10
dbConnect(), 3, 11
dbConnect,PrestoDriver-method (Presto), 9
dbCreateTableAs, 3
dbDataType,PrestoDriver-method, 4
dbDisconnect, 10
dbDisconnect,PrestoConnection-method
   (Presto), 9
dbGetInfo,PrestoConnection-method
   (dbGetInfo,PrestoDriver-method), 5
dbGetInfo,PrestoDriver-method, 5
dbGetInfo,PrestoResult-method
   (dbGetInfo,PrestoDriver-method), 5
DBI::dbConnect, 13
DBIConnection, 3
dbplyr::sql_query_save(), 12
dbplyr_edition.PrestoConnection, 6
dbQuoteIdentifier(), 3, 12
dbRenameTable, 6
Id(), 3, 12
kerberos_configs, 8
list(), 5, 10
logical(), 10
Presto, 9, 10
Presto(), 9
presto_default, 11
presto_has_default (presto_default), 11
PrestoConnection, 5, 10, 13
PrestoDriver, 4, 5, 10
PrestoResult, 5
SQL(), 3, 12
sql_query_save.PrestoConnection, 12
sqlCreateTableAs, 11
src_presto, 13