Package 'RTD'

July 27, 2020

Title Simple TD API Client

Version 0.4.1

Maintainer Aki Ariga <ariga@treasure-data.com>

Description Upload R data.frame to Arm Treasure Data, see <https://www.treasuredata.com/>. You can execute database or table handling for resources on Arm Treasure Data.

SystemRequirements embulk, embulk-output-td

License Apache License 2.0 | file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/treasure-data/RTD

BugReports https://github.com/treasure-data/RTD/issues

RoxygenNote 7.1.1

Collate 'TdClient.R' 'bulk_import.R' 'database.R' 'job.R' 'table.R' 'td.R'

Imports readr (>= 1.2.1), httr(>= 1.4.0), dplyr (>= 1.0.0), jsonlite, methods, RcppMsgPack, urltools, uuid, purrr

Suggests testthat (>= 2.0.1), mockery (>= 0.4.1.1), openssl, webmockr, devtools

NeedsCompilation no

Author Aki Ariga [aut, cre, cph]

Repository CRAN

Date/Publication 2020-07-26 23:10:22 UTC

R topics documented:

bulk_import_delete_part 2
bulk_import_error_records 3
bulk_import_upload_part 4
commit_bulk_import 4

create_bulk_import 5
create_database 6
create_table 6
delete_bulk_import 7
delete_database 8
delete_table 8
exist_database 9
exist_table 10
freeze_bulk_import 10
list_bulk_imports 11
list_bulk_import_parts 12
list_databases 12
list_tables 13
perform_bulk_import 14
show_bulk_import 14
Td 15
td_upload 16
unfreeze_bulk_import 17
update_schema 17
wait_bulk_import 18

Index **19**

`bulk_import_delete_part` *Delete bulk import part*

Description

Delete bulk import part

Usage

```
bulk_import_delete_part(conn, name, part_name)
```

Arguments

`conn`	`Td` client
`name`	Bulk import session name
`part_name`	Bulk import part name

Value

Return `TRUE` if succeeded

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
bulk_import_delete_part(conn, sess_name, "part")

## End(Not run)
```

`bulk_import_error_records` *Show bulk import error records*

Description

Show bulk import error records

Usage

```
bulk_import_error_records(conn, name)
```

Arguments

`conn`	`Td` client
`name`	Bulk import session name

Value

Return error records in gzipped file with msgpack stream format.

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
bulk_import_error_records(conn, sess_name)

## End(Not run)
```

bulk_import_upload_part *Upload bulk import part*

Description

Upload bulk import part

Usage

```
bulk_import_upload_part(conn, name, part_name, file_obj)
```

Arguments

conn	Td client
name	Bulk import session name
part_name	Bulk import part name
file_obj	File connection. Should be msgpack stream with gzip compressed. Should have "time" column

Value

Return bulk import status

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")

## End(Not run)
```

commit_bulk_import *Commit bulk import part*

Description

Commit bulk import part

Usage

```
commit_bulk_import(conn, name)
```

Arguments

conn	Td client
name	Bulk import session name

Value

Return TRUE if succeeded

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
commit_bulk_import(conn, sess_name)

## End(Not run)
```

---

create_bulk_import *Create bulk import*

Description

Create bulk import

Usage

```
create_bulk_import(conn, name, dbname, table)
```

Arguments

conn	Td client
name	Bulk import session name
dbname	Data base name
table	Table name

Value

Return TRUE if succeeded

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
sess_name <- uuid::UUIDgenerate()
create_bulk_import(conn, sess_name, "mydb", "mytable")

## End(Not run)
```

create_database *Create a database*

Description

Create a database

Usage

```
create_database(conn, dbname, params)
```

Arguments

conn	Td client
dbname	Target data base name
params	Optional parameters

Value

Returns TRUE or FALSE, whether the execution succeeded or not.

Examples

```
## Not run:
con <- Td(apikey = "xxxxx")
create_database(con, "newdb")

## End(Not run)
```

create_table *Create a table*

Description

Create a table

Usage

```
create_table(conn, dbname, table)
```

Arguments

conn	Td connection
dbname	Data base name
table	Table name

Value

Returns TRUE or FALSE, whether the execution succeeded or not.

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
create_table(conn, "mydb", "new_table")

## End(Not run)
```

`delete_bulk_import` *Delete bulk import*

Description

Delete bulk import

Usage

```
delete_bulk_import(conn, name)
```

Arguments

conn	Td client
name	Bulk import session name

Value

Return TRUE if succeeded

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
delete_bulk_import(conn, sess_name)

## End(Not run)
```

delete_database *Delete a database*

Description

Delete a database

Usage

```
delete_database(conn, dbname)
```

Arguments

conn	Td client
dbname	Target data base name

Value

Returns TRUE or FALSE, whether the execution succeeded or not.

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
delete_database(conn, "mydb")

## End(Not run)
```

delete_table *Delete a table*

Description

Delete a table

Usage

```
delete_table(conn, dbname, table)
```

Arguments

conn	Td connection
dbname	Data base name
table	Table name

Value

Returns TRUE or FALSE, whether the execution succeeded or not.

Examples

```
## Not run:
conn <- Td(apikey = "xxxxx")
delete_table(conn, "mydb", "iris")

## End(Not run)
```

`exist_database` *Check table existence*

Description

Check table existence

Usage

```
exist_database(conn, dbname)
```

Arguments

conn	Td client
dbname	Data base name

Value

Return TRUE or FALSE, existence

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
exist_database(conn, "mydb")

## End(Not run)
```

exist_table *Check table existence*

Description

Check table existence

Usage

```
exist_table(conn, dbname, table)
```

Arguments

conn	Td connection
dbname	Data base name
table	Table name

Value

Returns TRUE or FALSE, existence.

Examples

```
## Not run:
conn <- Td(apikey = "xxxxx")
exist_table(conn, "mydb", "iris")

## End(Not run)
```

freeze_bulk_import *Freeze bulk import part*

Description

Freeze bulk import part

Usage

```
freeze_bulk_import(conn, name)
```

Arguments

conn	Td client
name	Bulk import session name

Value

Return TRUE if succeeded

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
freeze_bulk_import(conn, sess_name)

## End(Not run)
```

---

list_bulk_imports *List bulk imports*

Description

List bulk imports

Usage

```
list_bulk_imports(conn)
```

Arguments

conn	Td client

Value

Return bulk import list

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
list_bulk_import(conn)

## End(Not run)
```

list_bulk_import_parts *List bulk import parts*

Description

List bulk import parts

Usage

```
list_bulk_import_parts(conn, name)
```

Arguments

conn	Td client
name	Bulk import session name

Value

Return bulk import parts list

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
list_bulk_import_parts(conn, sess_name)

## End(Not run)
```

list_databases *Show database list*

Description

Show database list

Usage

```
list_databases(conn)
```

Arguments

conn	Td connection

Value

Returns a `data.frame` of the database list

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
list_databases(conn)

## End(Not run)
```

`list_tables` *Show list of tables*

Description

Show list of tables

Usage

```
list_tables(conn, dbname)
```

Arguments

`conn`	`Td` connection
`dbname`	Data base name. Optional, but highly recommended to prevent timeout.

Value

Returns a `data.frame` of a list of tables or `FALSE` if not exists.

Examples

```
## Not run:
conn <- Td(apikey = "xxxxx")
list_tables(conn, "mydb")

## End(Not run)
```

perform_bulk_import *Perform bulk import part*

Description

Perform bulk import part

Usage

```
perform_bulk_import(conn, name)
```

Arguments

conn	Td client
name	Bulk import session name

Value

Return TRUE if succeeded

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
perform_bulk_import(conn, sess_name)

## End(Not run)
```

show_bulk_import *Show bulk import*

Description

Show bulk import

Usage

```
show_bulk_import(conn, name)
```

Arguments

conn	Td client
name	Bulk import session name

Value

Return bulk import status

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
show_bulk_import(conn, sess_name)

## End(Not run)
```

Td *Connect to TD*

Description

Connect to TD

Usage

```
Td(endpoint, apikey, user_agent, headers, http_proxy = NULL)
```

Arguments

endpoint	Endpoint to TD API
apikey	API key for TD
user_agent	User-Agent as character. optional
headers	Default headres in a named character vector. optional
http_proxy	HTTP proxy setting. optional.

Examples

```
## Not run:
client <- Td(
  endpoint = "api.treasuredata.com",
  apikey = "xxxxxx",
  http_proxy = "http://user:pass@proxy.domain.com:8080/"
)

## End(Not run)
```

td_upload *Upload data.frame to TD*

Description

Upload data.frame to TD

Usage

```
td_upload(
  conn,
  dbname,
  table,
  df,
  mode = "bulk_import",
  embulk_dir,
  overwrite = FALSE,
  append = FALSE
)
```

Arguments

conn	Td connection
dbname	Target destination database name.
table	Target table name.
df	Input data.frame.
mode	Write mode. "bulk_import" or "embulk". Default: "bulk_import"
embulk_dir	Path to embulk. [optional]
overwrite	Flag for overwriting the table if exists. It doesn't overwrite database. This flag sets "replace" mode for embulk-output-td.
append	Flag for append data into the table if exists. It doesn't overwrite database. This flag sets "append" mode for embulk-output-td.

Examples

```
## Not run:
td_upload_embulk("mydb", "iris", iris)

# With overwrite option
td_upload_embulk("mydb", "iris", iris, overwrite = TRUE)

# With append option
td_upload_embulk("mydb", "iris", iris, append = TRUE)

# With overwrite option
td_upload_embulk("mydb", "iris", iris, "/path/to/embulk", overwrite = TRUE)
```

```
## End(Not run)
```

`unfreeze_bulk_import` *Unfreeze bulk import part*

Description

Unfreeze bulk import part

Usage

```
unfreeze_bulk_import(conn, name)
```

Arguments

`conn`	`Td` client
`name`	Bulk import session name

Value

Return `TRUE` if succeeded

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
unfreeze_bulk_import(conn, sess_name)

## End(Not run)
```

`update_schema` *Update schema of a table*

Description

Update schema of a table

Usage

```
update_schema(conn, dbname, table, schema)
```

Arguments

conn	Td connection
dbname	Data base name
table	Table name
schema	Schema of the table to be updated

Value

Returns TRUE or FALSE, whether the execution succeeded or not.

Examples

```
## Not run:
conn <- Td(apikey = "xxxxx")
s <- rbind(
c("sepal_length", "double", "sepal_length"),
c("sepal_width", "double", "sepal_width"),
c("petal_length", "double", "petal_length"),
c("petal_width", "double", "petal_width"),
c("species", "string", "species"))
udpate_schema(conn, "mydb", "iris", s)

## End(Not run)
```

---

wait_bulk_import *Wait bulk import until finished*

Description

Wait bulk import until finished

Usage

```
wait_bulk_import(conn, sess_name)
```

Arguments

conn	Td client
sess_name	Bulk import session name

Examples

```
## Not run:
conn <- Td(apikey = "xxxx")
wait_bulk_import(conn, sess_name)

## End(Not run)
```

Index

bulk_import_delete_part, 2
bulk_import_error_records, 3
bulk_import_upload_part, 4

commit_bulk_import, 4
create_bulk_import, 5
create_database, 6
create_table, 6

delete_bulk_import, 7
delete_database, 8
delete_table, 8

exist_database, 9
exist_table, 10

freeze_bulk_import, 10

list_bulk_import_parts, 12
list_bulk_imports, 11
list_databases, 12
list_tables, 13

perform_bulk_import, 14

show_bulk_import, 14

Td, 15
td_upload, 16

unfreeze_bulk_import, 17
update_schema, 17

wait_bulk_import, 18