Package 'datrProfile'

May 8, 2026

Type Package

Title Column Profile for Tables and Datasets

Version 0.1.0

Description Profiles datasets (collecting statistics and informative summaries about that data) on data frames and 'ODBC' tables: maximum, minimum, mean, standard deviation, nulls, distinct values, data patterns, data/format frequencies.

License GPL-3 | file LICENSE

URL https://github.com/avitaliano/datrProfile

BugReports https://github.com/avitaliano/datrProfile/issues

Encoding UTF-8

LazyData true

Suggests testthat

Imports odbc, dplyr, RSQLite

RoxygenNote 6.1.1

NeedsCompilation no

Author Arnaldo Vitaliano [aut, cre]

Maintainer Arnaldo Vitaliano <vitaliano@gmail.com>

Repository CRAN

Date/Publication 2019-08-02 09:20:05 UTC

Contents

`buildQueryColumnFrequency`
buildQueryColumnFrequency

Description

buildQueryColumnFrequency

Usage

```
buildQueryColumnFrequency(conn.info, ...)
```

Arguments

`conn.info`	Connection info created with `prepareConnection`
`...`	Other parameters

Value

query column, count(*) from table

`buildQueryColumnMetadata`
buildQueryColumnMetadata

Description

buildQueryColumnMetadata

Usage

```
buildQueryColumnMetadata(conn.info, ...)
```

Arguments

conn.info	Connection info created with prepareConnection
...	Other params

Value

query columns' metadata

---

buildQueryColumnStats *buildQueryColumnStats*

Description

buildQueryColumnStats

Usage

```
buildQueryColumnStats(conn.info, ...)
```

Arguments

conn.info	Connection info created with prepareConnection
...	Other parameters

Value

query count(distinct column) from table

---

buildQueryColumnStats.sqlite *buildQueryColumnStats.sqlite*

Description

buildQueryColumnStats.sqlite

Usage

```
## S3 method for class 'sqlite'
buildQueryColumnStats(conn.info, schema, table, column,
  query.filter, ...)
```

Arguments

`conn.info`	Connection info created with `prepareConnection`
`schema`	Table Schema
`table`	Table Name
`column`	Column profiled
`query.filter`	Filter applied to the profile
`...`	Other parameters

Value

query count(distinct column) from table

`buildQueryCountNull` *buildQueryCountNull*

Description

buildQueryCountNull

Usage

```
buildQueryCountNull(conn.info, ...)
```

Arguments

`conn.info`	Connection info created with `prepareConnection`
`...`	Other parameters

Value

query select count(*) where collumn is null

`buildQueryCountTotal` *buildQueryCountTotal*

Description

Count total rows from table.

Usage

```
buildQueryCountTotal(conn.info, ...)
```

Arguments

`conn.info`	Connection info created with `prepareConnection`
`...`	Other params

Value

query count(*) from table

`buildQueryProfileColumnFormatFrequency` *buildQueryProfileColumnFormatFrequency*

Description

buildQueryProfileColumnFormatFrequency

Usage

```
buildQueryProfileColumnFormatFrequency(conn.info, ...)
```

Arguments

`conn.info`	Connection info created with `prepareConnection`
`...`	Other parameters

Value

queries column format frequency from table

closeConnection *closeConnection*

Description

Disconnects from database using odbc::dbDisconnect

Usage

```
closeConnection(conn)
```

Arguments

conn	Connection created at connectDB

Value

TRUE if succeeded at closing connection

connectDB *connectDB*

Description

Connects to database using dbConnect

Usage

```
connectDB(conn.info, ...)
```

Arguments

conn.info	Connection info created at prepareConnection
...	Other parameters

Value

connection to database

connectDB.default *connectDB.default*

Description

Connects to database using dbConnect

Usage

```
## Default S3 method:
connectDB(conn.info, ...)
```

Arguments

conn.info	Connection info created at prepareConnection
...	Other parameters

Value

connection to database

connectDB.sqlite *connectDB.sqlite*

Description

Connects to database using dbConnect

Usage

```
## S3 method for class 'sqlite'
connectDB(conn.info, ...)
```

Arguments

conn.info	Connection info created at prepareConnection
...	Other parameters

Value

connection to database

getTableColumns *getTableColumns*

Description

Issues query against the RDBS to retrieve information about each column of the table. Name, type, length, precision, etc.

Usage

```
getTableColumns(conn.info, schema, table)
```

Arguments

conn.info	Connection info created with prepareConnection
schema	Table schema
table	Table name

Value

data frame containing the columns' metadata

prepareConnection *Prepares connection to RDBS via ODBC*

Description

prepareConnection list connection details needed to connecto to a RDBS using ODBC

Usage

```
prepareConnection(db.vendor, odbc.driver = odbc::odbc(),
  db.host = NULL, db.name = NULL, db.encoding = "", dsn = NULL,
  user = NULL, passwd = NULL)
```

Arguments

db.vendor	Database vendor (teradata, sqlserver)
odbc.driver	ODBC driver used to connect to database
db.host	Database hostname
db.name	Database name
db.encoding	Database encoding
dsn	Data source name
user	Username to connect to database
passwd	Password to connect to database

Examples

```
conn.info <- prepareConnection(db.vendor = "teradata",
   dsn = "ODBC_MYDB", user = "myuser", passwd = "mypasswd")
```

`print.profile` *Print method*

Description

Print method

Usage

```
## S3 method for class 'profile'
print(x, ...)
```

Arguments

`x`	profile object
`...`	other parameters

Value

printed profile

`profileColumn` *profileColumn*

Description

profileColumn

Usage

```
profileColumn(conn.info, schema, table, column, column.datatype,
  query.filter, limit.freq.values = 30, format.show.percentage)
```

Arguments

conn.info	Connection info created with prepareConnection
schema	Table schema
table	Table name
column	Column being profiled
column.datatype	Column datatype
query.filter	Filter applied before profile the column
limit.freq.values	Distinct values shown in frequency data frame
format.show.percentage	Threshold considered when showing formats' percentages

Value

columnProfile <- list(column, count.total, count.distinct, perc.distinct, count.null, perc.null, min.value, max.value, column.freq, format.freq = format.freq)

---

profileColumnFormat *profileColumnFormat*

Description

Profiles column based on its format, using masking strategy. X = char, 9 = digit, S = symbol

Usage

```
profileColumnFormat(conn.info, column, column.datatype, schema, table,
  count.total, query.filter, format.show.percentage)
```

Arguments

conn.info	Connection info created with prepareConnection
column	Column name that will be profiled
column.datatype	Column datatipe
schema	Table schema
table	Table name
count.total	Number of rows to be profiled
query.filter	Filter applied to the table, when profilling
format.show.percentage	Threshold considered when showing formats' percentages

Value

Data Frame containing columns' metadata

runProfile *Profile all columns from ODBC table or dataframe*

Description

Profiles tables and dataframes (collecting statistics and informative summaries about that data): max, min, avg, sd, nulls, distinct values, data patterns, data/format frequencies.

Usage

```
runProfile(conn.info, schema = NULL, table, is.parallel = TRUE,
  count.nodes, query.filter = NA, format.show.percentage = 0.03)
```

Arguments

conn.info	Connection info created with prepareConnection
schema	Table schema
table	Table name
is.parallel	Boolean that indicates if profile will run in parallel. Default TRUE.
count.nodes	Number of nodes used when is.parallel = TRUE
query.filter	Filter applied to the table, when profilling
format.show.percentage	Threshold considered when showing formats percentages

Value

profile results for the table/dataframe

summary.profile *Override summary function*

Description

Override summary function

Usage

```
## S3 method for class 'profile'
summary(object, ...)
```

Arguments

object	Profile object
...	other parameters

Value

data.frame with summary information

Index