Package 'MonetDB.R'

October 12, 2022

Version 2.0.0

Title Connect MonetDB to R

Author Mitchell Weggemans [aut, cre],
Hannes Muehleisen [aut],
Anthony Damico [aut],
Thomas Lumley [ctb]

Maintainer Mitchell Weggemans <mitchell.weggemans@monetdbsolutions.com>

Imports DBI (>= 0.3.1), digest (>= 0.6.4), testthat, methods,
codetools

Description Allows to pull data from MonetDB into R.

License MPL (== 2.0)

URL http://www.monetdb.org

SystemRequirements MonetDB, available from http://www.monetdb.org

Collate mapi.R dbi.R

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-08-14 10:10:07 UTC

R topics documented:

dbSendUpdate *Send a data-altering SQL statement to the database.*

Description

dbSendUpdate is used to send a data-altering statement to a MonetDB database, e.g. CREATE TABLE or INSERT. As a convenience feature, a placeholder (? character) can be used in the SQL statement, and bound to parameters given in the varargs group before execution. This is especially useful when scripting database updates, since the parameters will be automatically quoted.

The dbSendUpdateAsync function is used when the database update is called from finalizers, to avoid very esoteric concurrency problems. Here, the update is not guaranteed to be immediately run. Also, the method returns immediately.

Usage

```
dbSendUpdate( conn, statement, ..., async=FALSE )
```

Arguments

conn	A MonetDB.R database connection. Created using dbConnect with the MonetDB.R database driver.
statement	A SQL statement to be sent to the database, e.g. 'UPDATE' or 'INSERT'.
...	Parameters to be bound to '?' characters in the query, similar to JDBC.
async	Behave like dbSendUpdateAsync? Defaults to FALSE.

Value

Returns TRUE if the update was successful.

See Also

dbSendQuery

Examples

```
## Not run:
# connect to MonetDB
conn <- dbConnect(MonetDB.R(), "monetdb://localhost/acs")
# create table
dbSendUpdate(conn, "CREATE TABLE foo(a INT,b VARCHAR(100))")
# insert value, bind parameters to placeholders in statement
dbSendUpdate(conn, "INSERT INTO foo VALUES(?,?)", 42, "bar")


## End(Not run)
```

dbTransaction *Create, commit or abort a database transaction.*

Description

dbTransaction is used to switch the data from the normal auto-commiting mode into transactional mode. Here, changes to the database will not be permanent until dbCommit is called. If the changes are not to be kept around, you can use dbRollback to undo all the changes since dbTransaction was called.

Usage

```
dbTransaction(conn, ...)
```

Arguments

conn	A MonetDB.R database connection. Created using dbConnect with the MonetDB.R database driver.
...	Future use.

Value

Returns TRUE if the transaction command was successful.

Examples

```
## Not run:
conn <- dbConnect(MonetDB.R(), "monetdb://localhost/acs")
dbSendUpdate(conn, "CREATE TABLE foo(a INT,b VARCHAR(100))")
dbTransaction(conn)
dbSendUpdate(conn, "INSERT INTO foo VALUES(?,?)", 42, "bar")
dbCommit(conn)
dbTransaction(conn)
dbSendUpdate(conn, "INSERT INTO foo VALUES(?,?)", 43, "bar")
dbRollback(conn)

# only 42 will be in table foo

## End(Not run)
```

MonetDB.R *DBI database connector for MonetDB*

Description

MonetDB.R creates a new DBI driver that can be used to connect and interact with MonetDB.

Usage

```
MonetDB.R ()
```

Details

The MonetDB.R function creates the R object which can be used to a call dbConnect which actually creates the connection. Since it has no parameters, it is most commonly used inline with the dbConnect call.

This package aims to provide a reasonably complete implementation of the DBI. A number of additional methods are provided: dbSendUpdate for database-altering statements, dbSendUpdateAsync for cleanup operations and monetdb.read.csv for database CSV import.

Value

Returns a driver object that can be used in calls to dbConnect.

See Also

dbConnect for documentation how to invoke the driver monetdb.server.setup to set up and start a local MonetDB server from R

Examples

```
## Not run:
library(DBI)
conn <- dbConnect(MonetDB.R::MonetDB(), dbname = "demo")
dbWriteTable(conn, "iris", iris)
dbListTables(conn)
dbGetQuery(conn, "SELECT COUNT(*) FROM iris")
d <- dbReadTable(conn, "iris")

## End(Not run)
```

`monetdb.read.csv` *Import a CSV file into MonetDB*

Description

Instruct MonetDB to read a CSV file, optionally also create the table for it.

Usage

```
monetdb.read.csv (conn, files, tablename, header=TRUE,
best.effort=FALSE, delim=",",
newline = "\\n", quote = "\"", create=TRUE, col.names=NULL, lower.case.names=FALSE,
sep=delim, ...)
```

Arguments

`conn`	A MonetDB.R database connection. Created using `dbConnect` with the `MonetDB.R` database driver.
`files`	A single string or a vector of strings containing the absolute file names of the CSV files to be imported.
`tablename`	Name of the database table the CSV files should be imported in. Created if necessary.
`header`	Whether or not the CSV files contain a header line.
`best.effort`	Use best effort flag when reading csv files and continue importing even if parsing of fields/lines fails.
`delim`	Field separator in CSV file.
`newline`	Newline in CSV file, usually \n for UNIX-like systems and \r\r on Windows.
`quote`	Quote character(s) in CSV file.
`create`	Create table before importing?
`lower.case.names`	Convert all column names to lowercase in the database?
`col.names`	Optional column names in case the ones from CSV file should not be used
`sep`	alias for `delim`
`...`	Additional parameters. Currently not in use.

Value

Returns the number of rows imported if successful.

See Also

`dbWriteTable` in `DBIConnection-class`

Examples

```
## Not run:
library(DBI)
# connect to MonetDB
conn <- dbConnect(MonetDB.R::MonetDB(), dbname = "demo")
# write test data to temporary CSV file
file <- tempfile()
write.table(iris, file, sep=",")
# create table and import CSV
MonetDB.R::monetdb.read.csv(conn, file, "iris")

## End(Not run)
```

---

monetdbRtype *Get the name of the R data type for a database type.*

Description

For a database data type, get the name of the R data type it is being translated to.

Usage

```
monetdbRtype ( dbType )
```

Arguments

dbType	A database type string such as `CHAR` or `INTEGER`.

Value

String containing the R data type for the DB data type, e.g. `character` or `numeric`.

---

sqlite-compatibility *Compatibility functions for RSQlite*

Description

Some functions that RSQlite has and that we support to allow MonetDBLite being used as a drop-in replacement.

Usage

```
isIdCurrent(dbObj, ...)
initExtension(dbObj, ...)
```

Arguments

db0bj	A MonetDB.R database connection. Created using dbConnect with the MonetDB.R database driver.
...	Additional parameters. Currently not in use.

Index