

Package ‘neo2R’

April 15, 2022

Type Package

Title Neo4j to R

Version 2.1.1

Maintainer Patrice Godard <patrice.godard@gmail.com>

Description The aim of the neo2R is to provide simple and low level connectors for querying neo4j graph databases (<<https://neo4j.com/>>). The objects returned by the query functions are either lists or data.frames with very few post-processing. It allows fast processing of queries returning many records. And it let the user handle post-processing according to the data model and his needs.

URL <https://github.com/patzaw/neo2r>

BugReports <https://github.com/patzaw/neo2r/issues>

Depends R (>= 3.6)

Imports base64enc, jsonlite, RCurl, utils

SystemRequirements neo4j (==3 OR ==4) <<https://neo4j.com/>>

License GPL-3

Encoding UTF-8

RoxygenNote 7.1.2

NeedsCompilation no

Author Patrice Godard [aut, cre, cph]

Repository CRAN

Date/Publication 2022-04-15 13:30:02 UTC

R topics documented:

cypher	2
graphRequest	3
import_from_df	4
multicypher	4

prepCql	6
readCql	6
startGraph	7

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

cypher	<i>Run a cypher query</i>
--------	---------------------------

Description

Run a cypher query

Usage

```
cypher(
  graph,
  query,
  parameters = NULL,
  result = c("row", "graph"),
  arraysAsStrings = TRUE,
  eltSep = " || "
)
```

Arguments

graph	the neo4j connection
query	the cypher query
parameters	parameters for the cypher query.
result	the way to return results. "row" will return a data frame and "graph" will return a list of nodes, a list of relationships and a list of paths (vectors of relationships identifiers).
arraysAsStrings	if result="row" and arraysAsStrings is TRUE (default) array from neo4j are converted to strings and array elements are separated by eltSep.
eltSep	if result="row" and arraysAsStrings is TRUE (default) array from neo4j are converted to strings and array elementes are separated by eltSep.

Value

the "result" of the query (invisible). See the "result" param.

See Also

[multicypher\(\)](#), [startGraph\(\)](#), [prepCql\(\)](#), [readCql\(\)](#) and [graphRequest\(\)](#)

Examples

```
## Not run:
# 2 identical queries
result <- cypher(
  graph=graph,
  query='match (n {value:$value}) return n',
  parameters=list(value="100"),
  result="graph"
)
result <- cypher(
  graph=graph,
  query='match (n {value:"100"}) return n',
  result="graph"
)

## End(Not run)
```

graphRequest

Run a curl request on a neo4j graph

Description

Run a curl request on a neo4j graph

Usage

```
graphRequest(graph, endpoint, customrequest = c("POST", "GET"), postText)
```

Arguments

graph	the neo4j connection
endpoint	the endpoint for the request. To list all the available endpoints: graphRequest(graph, endpoint="", cus
customrequest	the type of request: "POST" (default) or "GET"
postText	the request body

Value

a list with the "header" and the "result" of the request (invisible)

See Also

[startGraph\(\)](#) and [cypher\(\)](#)

import_from_df	<i>Imports a data.frame in the neo4j graph database</i>
----------------	---

Description

This function only works with localhost Neo4j instances.

Usage

```
import_from_df(graph, cql, toImport, periodicCommit = 10000, ...)
```

Arguments

graph	the neo4j connection
cql	the CQL query to be applied on each row of toImport. Use the 'row' prefix to refer to the data.frame column.
toImport	the data.frame to be imported as "row". Use "row.FIELD" in the cql query to refer to one FIELD of the toImport data.frame
periodicCommit	use periodic commit when loading the data (default: 10000).
...	further parameters for cypher()

See Also

[cypher\(\)](#)

multicypher	<i>Run a multiple cypher queries</i>
-------------	--------------------------------------

Description

Run a multiple cypher queries

Usage

```
multicypher(
  graph,
  queries,
  parameters = NULL,
  result = c("row", "graph"),
  arraysAsStrings = TRUE,
  eltSep = " || "
)
```

Arguments

graph	the neo4j connection
queries	queries to submit. It can be either a character vector for which each element corresponds to a cypher query. Or it can be a list of lists with the following slots: <ul style="list-style-type: none"> • query (mandatory): A single character corresponding to the cypher query. • parameters (optional): A set of parameters specific for this query. If not provided, the <i>parameters</i> parameter of the function is used (see below). • result (optional): The specific way to return the results of this query. If not provided, the <i>result</i> parameter of the function is used (see below).
parameters	default parameters for the cypher queries.
result	default way to return results. "row" will return a data frame and "graph" will return a list of nodes, a list of relationships and a list of paths (vectors of relationships identifiers).
arraysAsStrings	if result="row" and arraysAsStrings is TRUE (default) array from neo4j are converted to strings and array elements are separated by eltSep.
eltSep	if result="row" and arraysAsStrings is TRUE (default) array from neo4j are converted to strings and array elementes are separated by eltSep.

Value

a list of "result" of the queries (invisible). See the "result" param.

See Also

[cypher\(\)](#), [startGraph\(\)](#), [prepCql\(\)](#), [readCql\(\)](#) and [graphRequest\(\)](#)

Examples

```
## Not run:
result <- multicypher(
  graph,
  queries=list(
    q1="match (n) return n.value limit 5",
    q2=list(
      query="match (f {value:$val})-[r]->(t) return f, r, t limit 5",
      result="graph",
      parameters=list(val=100)
    )
  )
)
## End(Not run)
```

prepCql

Prepares a CQL query from a character vector

Description

Prepares a CQL query from a character vector

Usage

```
prepCql(...)
```

Arguments

... character vectors with cQL commands

Value

a well formatted CQL query

See Also

[cypher\(\)](#) and [readCql\(\)](#)

Examples

```
prepCql(c(  
  "MATCH (n)",  
  "RETURN n"  
))
```

readCql

Parse a CQL file and returned the prepared queries

Description

Parse a CQL file and returned the prepared queries

Usage

```
readCql(file)
```

Arguments

file the name of the file to be parsed

Value

a character vector of well formatted CQL queries

See Also

[cypher\(\)](#) and [prepCql\(\)](#)

startGraph	<i>Prepare connection to neo4j database</i>
------------	---

Description

Prepare connection to neo4j database

Usage

```
startGraph(
  url,
  database = NA,
  username = NA,
  password = NA,
  importPath = NA,
  .opts = list()
)
```

Arguments

url	the DB url
database	the name of the database. If NA (default) it will use "data" with versions 3.. of Neo4j and "neo4j" with versions 4..
username	the neo4j user name (default: NA; works only if authentication has been disabled in neo4j by setting NEO4J.AUTH=none)
password	the neo4j user password (default: NA; works only if authentication has been disabled in neo4j by setting NEO4J.AUTH=none)
importPath	path to the import directory (default: NA => no import directory). Import only works with local neo4j instance.
.opts	a named list or CURLOptions object identifying the curl options for the handle (see RCurl::curlPerform()). (for example: .opts = list(ssl.verifypeer = FALSE))

Value

a connection to the graph DB: a list with the url and necessary headers

Index

cypher, [2](#)
cypher(), [3–7](#)

graphRequest, [3](#)
graphRequest(), [2, 5](#)

import_from_df, [4](#)

multicypher, [4](#)
multicypher(), [2](#)

prepCql, [6](#)
prepCql(), [2, 5, 7](#)

RCurl::curlPerform(), [7](#)
readCql, [6](#)
readCql(), [2, 5, 6](#)

startGraph, [7](#)
startGraph(), [2, 3, 5](#)