Package 'neo4r'

April 3, 2019

Title A 'Neo4J' Driver

Version 0.1.1

Description A Modern and Flexible 'Neo4J' Driver, allowing you to query data on a 'Neo4J' server and handle the results in R. It's modern in the sense it provides a driver that can be easily integrated in a data analysis workflow, especially by providing an API working smoothly with other data analysis and graph packages. It's flexible in the way it returns the results, by trying to stay as close as possible to the way 'Neo4J' returns data. That way, you have the control over the way you will compute the results. At the same time, the result is not too complex, so that the ``heavy lifting'' of data wrangling is not left to the user.

License MIT + file LICENSE

URL `https://github.com/neo4j-rstats/neo4r`

BugReports `https://github.com/neo4j-rstats/neo4r/issues`

Imports attempt, data.table, glue, httr, igraph, jsonlite, magrittr, purrr, R6, rlang, rstudioapi, shiny, tibble, tidyr, tidyselect, utils

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

NeedsCompilation no

Author Colin Fay [cre, aut] (¡https://orcid.org/0000-0001-7343-1846¿),
ThinkR [cph],
Neo4J [spn]

Maintainer Colin Fay <`contact@colinfay.me`>

Repository CRAN

Date/Publication 2019-02-15 22:20:03 UTC

R topics documented:

call_neo4j *Call Neo4J API*

Description

Call Neo4J API

Usage

```
call_neo4j(query, con, type = c("row", "graph"), output = c("r",
  "json"), include_stats = FALSE, include_meta = FALSE)
```

Arguments

`query`	The cypher query
`con`	A NEO4JAPI connection object
`type`	Return the result as row or as graph
`output`	Use ”json” if you want the output to be printed as JSON
`include_stats`	tShould the stats about the transaction be included?
`include_meta`	tShould the stats about the transaction be included?

Value

the result from the Neo4J Call

extract_nodes *Extract nodes or relationships*

Description

Extract nodes or relationships

Usage

```
extract_nodes(x)

extract_relationships(x)
```

Arguments

`x`	a result from Neo4J

Value

a tibble

`launch_con_pane` *Launch Neo4J Connection Pane*

Description

Launch Neo4J Connection Pane

Usage

```
launch_con_pane(con)
```

Arguments

`con`	a connection object

Value

an opened Connection Pane

`load_csv` *Load a CSV to Neo4J*

Description

Load a CSV to Neo4J

Usage

```
load_csv(on_load = "", con, url, header = TRUE,
  periodic_commit = 1000, as = "csv", type = c("row", "graph"),
  output = c("r", "json"), include_stats = TRUE,
  include_meta = FALSE)
```

Arguments

`on_load`	the code to execute on load
`con`	A NEO4JAPI connection object
`url`	the url of the csv
`header`	does the csv have a header?
`periodic_commit`	the PERIODIC COMMIT cypher arg
`as`	the AS cypher arg

`type`	Return the result as row or as graph
`output`	Use ”json” if you want the output to be printed as JSON
`include_stats`	tShould the stats about the transaction be included?
`include_meta`	tShould the stats about the transaction be included?

Value

a csv loaded to Neo4J

`neo4j_api` *A Neo4J Connexion*

Description

A Neo4J Connexion

Usage

```
neo4j_api
```

Format

An object of class `R6ClassGenerator` of length 24.

Value

A Neo4J Connexion

Methods

`access` list url, user and password

`ping` test your connexion

`version` Neo4J version

`get` Get a list of either relationship, labels,

`get` Get a list of either relationship, labels,

`get` Get a list of either relationship, labels,

`get` Get a list of either relationship, labels,

`get` Get a list of either relationship, labels,

Data

`url` list url, user and password

`user` test your connexion

Examples

```
## Not run:
con <- neo4j_api$new(url = "http://localhost:7474", user = "neo4j", password = "password")

## End(Not run)
```

read_cypher *Read a cypher file*

Description

Read a cypher file

Usage

```
read_cypher(file)
```

Arguments

file	the path to the cypher file

Value

a tibble with the queries

Examples

```
## Not run:
read_cypher("random/create.cypher")

## End(Not run)
```

send_cypher *Send a cypher file to be executed*

Description

Send a cypher file to be executed

Usage

```
send_cypher(path, con, type = c("row", "graph"), output = c("r",
  "json"), include_stats = TRUE, meta = FALSE)
```

Arguments

path	the path to the cypher file
con	a connexion object created with neo4j_api$new()
type	the type of the format to query for (row or graph)
output	the printing method (r or json)
include_stats	whether of not to include stats
meta	whether of not to include meta info

Value

a cypher call

Examples

```
## Not run:
send_cypher("random/create.cypher")
path <- "data-raw/constraints.cypher"

## End(Not run)
```

unnest_graph *Unnest both relationships and nodes*

Description

Unnest both relationships and nodes

Usage

```
unnest_graph(res)
```

Arguments

`res`	an api graph result

Value

a list of two unnested data.frames

unnest_nodes *Unnest a node data.frame*

Description

Unnest a node data.frame

Usage

```
unnest_nodes(nodes_tbl, what = c("all", "label", "properties"))
```

Arguments

`nodes_tbl`	the node table
`what`	what to unnest

Value

a new dataframe

unnest_relationships *Unnest a Relationships table*

Description

Unnest a Relationships table

Usage

```
unnest_relationships(relationships_tbl)
```

Arguments

`relationships_tbl`
a relationship table

Value

an unnested table

Note

Please note that the properties will be converted to character if the class is not unique.

vec_to_cypher *Turn a named vector into a cypher list*

Description

'vec_to_cypher()' creates a list, and 'vec_to_cypher_with_var()' creates a cypher call starting with a variable.

Usage

```
vec_to_cypher(vec, label)

vec_to_cypher_with_var(vec, label, variable)
```

Arguments

`vec`	the vector
`label`	the label of each vector
`variable`	the variable to use (for 'vec_to_cypher()')

Details

This function can be used with small vectors you want to send to the server. It can for example be used this way : "' paste("MERGE", vec_to_cypher(iris[1, 1:3], "Species")) "' to create a cypher call.

Value

a character vector

Examples

```
vec_to_cypher(iris[1, 1:3], "Species")
vec_to_cypher_with_var(iris[1, 1:3], "Species", a)
```