

# Package: neo4r (via r-universe)

June 11, 2024

**Title** A 'Neo4J' Driver

**Version** 0.1.3

**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 (>= 0.3.2), R6, rlang, rstudioapi, shiny, tibble, tidyr, tidyselect, utils

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.1.0

**Repository** <https://neo4j-rstats.r-universe.dev>

**RemoteUrl** <https://github.com/neo4j-rstats/neo4r>

**RemoteRef** HEAD

**RemoteSha** d23198dc7ba0d6a4805a29456770d0df7ea28f82

## Contents

call_neo4j . . . . .	2
extract_nodes . . . . .	3

launch_con_pane . . . . .	3
load_csv . . . . .	4
neo4j_api . . . . .	4
play_movies . . . . .	5
read_cypher . . . . .	6
send_cypher . . . . .	6
unnest_graph . . . . .	7
unnest_nodes . . . . .	7
unnest_relationships . . . . .	8
vec_to_cypher . . . . .	8

<b>Index</b>	<b>10</b>
--------------	-----------

---

call_neo4j	<i>Call Neo4J API</i>
------------	-----------------------

---

## 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	<i>Extract nodes or relationships</i>
---------------	---------------------------------------

---

**Description**

Extract nodes or relationships

**Usage**

extract\_nodes(x)

extract\_relationships(x)

**Arguments**

x                    a result from Neo4J

**Value**

a tibble

---

launch_con_pane	<i>Launch Neo4J Connection Pane</i>
-----------------	-------------------------------------

---

**Description**

Launch Neo4J Connection Pane

**Usage**

launch\_con\_pane(con)

**Arguments**

con                    a connection object

**Value**

an opened Connection Pane

---

load_csv	<i>Load a CSV to Neo4J</i>
----------	----------------------------

---

**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	<i>A Neo4J Connexion</i>
-----------	--------------------------

---

**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)
```

---

play\_movies

*The Movie Graph*

---

**Description**

The Movie Graph

**Usage**

```
play_movies()
```

**Value**

A character vector with the movie db

---

read_cypher	<i>Read a cypher file</i>
-------------	---------------------------

---

**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	<i>Send a cypher file to be executed</i>
-------------	--

---

**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	<i>Unnest both relationships and nodes</i>
--------------	--

---

**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	<i>Unnest a node data.frame</i>
--------------	---------------------------------

---

**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)
```

# Index

## \* datasets

- neo4j\_api, 4
- call\_neo4j, 2
- extract\_nodes, 3
- extract\_relationships (extract\_nodes), 3
- launch\_con Pane, 3
- load\_csv, 4
- neo4j\_api, 4
- play\_movies, 5
- read\_cypher, 6
- send\_cypher, 6
- unnest\_graph, 7
- unnest\_nodes, 7
- unnest\_relationships, 8
- vec\_to\_cypher, 8
- vec\_to\_cypher\_with\_var (vec\_to\_cypher),  
8