Package ‘ReactomeGraph4R’

March 19, 2024

Title Interface for the Reactome Graph Database
Version 1.10.0
Description Pathways, reactions, and biological entities in Reactome knowledge are systematically represented as an ordered network. Instances are represented as nodes and relationships between instances as edges; they are all stored in the Reactome Graph Database. This package serves as an interface to query the interconnected data from a local Neo4j database, with the aim of minimizing the usage of Neo4j Cypher queries.
License Apache License (>= 2)
Encoding UTF-8
URL https://github.com/reactome/ReactomeGraph4R
BugReports https://github.com/reactome/ReactomeGraph4R/issues
Roxygen list(markdown = TRUE)
RoxygenNote 7.1.1
Depends R (>= 4.1)
Imports neo4r, utils, getPass, jsonlite, purrr, magrittr, data.table, rlang, ReactomeContentService4R, doParallel, parallel, foreach
Suggests knitr, rmarkdown, testthat, stringr, networkD3, visNetwork, wesanderson
VignetteBuilder knitr
bioViews DataImport, Pathways, Reactome, Network, GraphAndNetwork
git_url https://git.bioconductor.org/packages/ReactomeGraph4R
git_branch RELEASE_3_18
git_last_commit 01c5bce
git_last_commit_date 2023-10-24
Repository Bioconductor 3.18
Date/Publication 2024-03-18
Author Chi-Lam Poon [aut, cre] (<https://orcid.org/0000-0001-6298-7099>), Reactome [cph]
Maintainer Chi-Lam Poon <clpoon807@gmail.com>
Description

Pathways, reactions, and biological entities in Reactome knowledge are systematically represented as an ordered network. Instances are represented as nodes and relationships between instances as edges; they are all stored in the Reactome Graph Database. This package serves as an interface to query the interconnected data from a local Neo4j database, with the aim of minimizing the usage of Neo4j Cypher queries.

Author(s)

Maintainer: Chi-Lam Poon <clpoon807@gmail.com> (ORCID)

Other contributors:

- Reactome <help@reactome.org> [copyright holder]

See Also

Useful links:

- https://github.com/reactome/ReactomeGraph4R
- Report bugs at https://github.com/reactome/ReactomeGraph4R/issues
**login**  
*Log in to the local neo4j server*

**Description**

Before running `login()`, you have to successfully finish the Reactome Neo4j database setup and build a connection on your local machine (details see: https://github.com/reactome/ReactomeGraph4R). This command is to create a neo4r object that is used to communicate between R and Neo4j, also to do a sanity check for the connection.

**Usage**

```r
login(con = NULL)
```

**Arguments**

- `con`  
an existed connexion object. It is not necessary to log in for the first time.

**Value**

connection to the local neo4j database

**Examples**

```r
## Not run:
# The first step to the graph database!
login()

## End(Not run)
# you can also check the neo4r connexion object by running:
getOption("con")
```

---

**matchDiseases**  
*MATCH diseases of PhysicalEntity/Reaction/Pathway*

**Description**

To find Diseases related to a PhysicalEntity or an Event, or get PhysicalEntities/Events associated with a Disease in reverse

**Usage**

```r
matchDiseases(
  id = NULL,
  displayName = NULL,
  species = NULL,
  type = c("row", "graph")
)
```
### Arguments

- **id**: stId or dbId of a PhysicalEntity/Event/Disease
- **displayName**: displayName of a PhysicalEntity/Event/Disease
- **species**: name or taxon id or dbId or abbreviation of aspecies
- **type**: return results as a list of dataframes ('row'), or as a graph object ('graph')

### Value

Disease(s) related to the given PhysicalEntity/Reaction/Pathway; or instances related to the given Disease

### See Also

Other match: `matchHierarchy()`, `matchInteractors()`, `matchObject()`, `matchPEroles()`, `matchPaperObjects()`, `matchPrecedingAndFollowingEvents()`, `matchReactionsInPathway()`, `matchReferrals()`

### Examples

```r
disease <- "neuropathy"
# matchDiseases(displayName=disease, species="M. musculus", type="row")
# matchDiseases(id="R-HSA-162588", type="graph")
```

---

**matchHierarchy**

### Description

Reactome data are organized in a hierarchical way: Pathway-Reaction-Entity. This function retrieves the hierarchical data of a given Event (Pathway or Reaction) or Entity (PhysicalEntity or ReferenceEntity).

### Usage

```r
matchHierarchy(
  id = NULL,
  displayName = NULL,
  databaseName = "Reactome",
  species = NULL,
  type = c("row", "graph")
)
```

### Arguments

- **id**: stId or dbId of an Event/Entity; or an external id
- **displayName**: displayName of Event/PhysicalEntity/ReferenceEntity
- **databaseName**: database name
- **species**: name or taxon id or dbId or abbreviation of specified species
- **type**: return results as a list of dataframes ('row'), or as a graph object ('graph')
### Description

To retrieve interactions of a given PhysicalEntity (PE), it first finds the ReferenceEntity matched with the PE, then get the Interactions having "interactor" relationship with the ReferenceEntity.

### Usage

```r
matchInteractors(
  pe.id = NULL,
  pe.displayName = NULL,
  species = NULL,
  type = c("row", "graph")
)
```

### Arguments

- **pe.id**: stId or dbId of a PhysicalEntity
- **pe.displayName**: displayName of a PhysicalEntity
- **species**: name or taxon id or dbId or abbreviation of specified species
- **type**: return results as a list of dataframes ('row'), or as a graph object ('graph')

### Value

interactions of a given PhysicalEntity

### See Also

Other match: `matchDiseases()`, `matchHierarchy()`, `matchObject()`, `matchPEroles()`, `matchPaperObjects()`, `matchPrecedingAndFollowingEvents()`, `matchReactionsInPathway()`, `matchReferrals()`
matchObject

Basic query for database objects

Description

This function can fetch instance by setting the following arguments:

- **id**: a Reactome dbId/stId, or non-Reactome id (e.g. UniProt)
- **displayName**: a display name of a Reactome object
- **schemaClass**: a specific schema class, see Data Schema
- **property**: a property of a node or relationship, access the full list of properties: con <- getOption("con"); con$get_property_keys()
- **relationship**: a relationship between nodes, access the full list of relationships: con <- getOption("con"); con$get_relationships()
- **Species information can see here**, or run View(matchObject(schemaClass = "Species")[["databaseObject"]]) to view a full table

Usage

```r
matchObject(
  id = NULL,
  displayName = NULL,
  schemaClass = NULL,
  species = NULL,
  returnedAttributes = NULL,
  property = NULL,
  relationship = NULL,
  limit = NULL,
  databaseName = "Reactome"
)
```

Arguments

- **id**: Reactome stId or dbId, or non-Reactome identifier
- **displayName**: display name of a database object
- **schemaClass**: schema class of a database object
- **species**: name or taxon id or dbId or abbreviation of specified species
- **returnedAttributes**: specific attribute(s) to be returned. If set to NULL, all attributes returned
- **property**: a list of property keys and values, e.g. list(isChimeric = TRUE, isInDisease = TRUE)
- **relationship**: relationship type(s)
- **limit**: the number of returned objects
- **databaseName**: database name. All databases see here
**matchPaperObjects**

**Value**
Reactome database object(s) that meets all specified conditions

**See Also**
- `multiObjects` for multiple ids
- Other match: `matchDiseases()`, `matchHierarchy()`, `matchInteractors()`, `matchPEroles()`, `matchPaperObjects()`, `matchPrecedingAndFollowingEvents()`, `matchReactionsInPathway()`, `matchReferrals()`

**Examples**

```r
## fetch instance by class
# all.species <- matchObject(schemaClass = "Species")

## fetch instance by name
# matchObject(displayName = "RCOR1 [nucleoplasm]",
# returnedAttributes=c("stId", "speciesName"))

## fetch instance by id
## Reactome id
# matchObject(id = "R-HSA-9626034")
## non-Reactome id
# matchObject(id = "P60484", databaseName = "UniProt")

## fetch instances by relationship
# matchObject(relationship="inferredTo", limit=10)

## fetch instances by property
property.list <- list(hasEHLD = TRUE, isInDisease = TRUE)
# matchObject(property = property.list,
# returnedAttributes = c("displayName", "stId", "isInDisease", "hasEHLD"),
# limit=20)
```

**matchPaperObjects**  
MATCH objects related to a paper

**Description**
Fetch Reactome instances related to a paper by its PubMed id or title

**Usage**

```r
matchPaperObjects(
  pubmed.id = NULL,
  displayName = NULL,
  type = c("row", "graph")
)
```
**matchPEroles**

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pubmed.id</td>
<td>PubMed identifier of a paper</td>
</tr>
<tr>
<td>displayName</td>
<td>paper title</td>
</tr>
<tr>
<td>type</td>
<td>return results as a list of dataframes ('row'), or as a graph object ('graph')</td>
</tr>
</tbody>
</table>

**Value**

Reactome instances associated with a paper

**See Also**

Other match: `matchDiseases()`, `matchHierarchy()`, `matchInteractors()`, `matchObject()`, `matchPEroles()`, `matchPrecedingAndFollowingEvents()`, `matchReactionsInPathway()`, `matchReferrals()`

**Examples**

```r
## fetch Reactome instances by paper title
paper <- "Chaperone-mediated autophagy at a glance"
# matchPaperObjects(displayName=paper)

## fetch Reactome instances by pubmed id
# matchPaperObjects(pubmed.id="20797626", type="graph")
# matchPaperObjects(pubmed.id="23515720", type="row")
```

**Description**

This function retrieves the role(s) of a given PhysicalEntity including:

- Input
- Output
- Regulator
- Catalyst

**Usage**

```r
matchPEroles(
  pe.id = NULL,
  pe.displayName = NULL,
  species = NULL,
  type = c("row", "graph")
)
```
matchPrecedingAndFollowingEvents

Arguments

pe.id  stId or dbId of a PhysicalEntity
pe.displayName  displayName of a PhysicalEntity
species  name or taxon id or dbId or abbreviation of a species
type  return results as a list of dataframes ('row'), or as a graph object ('graph')

Value

information of the given PhysicalEntity and its role(s)

See Also

Other match: matchDiseases(), matchHierarchy(), matchInteractors(), matchObject(), matchPaperObjects(), matchPrecedingAndFollowingEvents(), matchReactionsInPathway(), matchReferrals()

Examples

stId <- "R-HSA-8944354"
# matchPEroles(pe.id = stId, type = "graph")

# matchPEroles(pe.displayName = "2SUMO1:MITF [nucleoplasm]",
#               species = "pig", type = "row")

matchPrecedingAndFollowingEvents

MATCH the preceding/following Events

Description

This method can find preceding and following ReactionLikeEvents (RLEs) of a specific Event with the relationship 'precedingEvent'. The argument "depth" is used to describe the "variable length relationships" in Neo4j, default is 1 (i.e. immediately connected); or you can set all.depth = TRUE to retrieve the whole context.

Usage

matchPrecedingAndFollowingEvents(
  event.id = NULL,
  event.displayName = NULL,
  species = NULL,
  depth = 1,
  all.depth = FALSE,
  type = c("row", "graph")
)
Arguments

- **event.id**: stId/dbId of an Event
- **event.displayName**: displayName of an Event
- **species**: name or taxon id or dbId or abbreviation of specified species
- **depth**: number of depths
- **all.depth**: if set to TRUE, all RLE(s) connected to the given Event in all depths returned
- **type**: to return results as a list of dataframes ('row'), or as a graph object ('graph')

Value

preceding/following Events connected to the given Event in specified depth(s), default depth = 1

See Also

Other match: `matchDiseases()`, `matchHierarchy()`, `matchInteractors()`, `matchObject()`, `matchPEroles()`, `matchPaperObjects()`, `matchReactionsInPathway()`, `matchReferrals()`

Examples

```r
stId <- "R-HSA-983150"
# matchPrecedingAndFollowingEvents(event.id=stId, depth=2, type="row")
```

Description

This method could find all Reactions connected with a given Pathway by the relationship 'hasEvent'. Also, the input can be a Reaction, the result would then be Pathway(s) linked via 'hasEvent' together with other Reactions linked with the Pathways(s).

Usage

```r
matchReactionsInPathway(
  event.id = NULL,
  event.displayName = NULL,
  species = NULL,
  type = c("row", "graph")
)
```
matchReferrals

Arguments

- **event.id**: stId or dbId of an Event
- **event.displayName**: displayName of an Event
- **species**: name or taxon id or dbId or abbreviation of a species
- **type**: return results as a list of dataframes ('row'), or as a graph object ('graph')

Value

Reactions connected to the given Pathway/Reaction via 'hasEvent' relationships

See Also

Other match: *matchDiseases()*, *matchHierarchy()*, *matchInteractors()*, *matchObject()*, *matchPEroles()*, *matchPaperObjects()*, *matchPrecedingAndFollowingEvents()*, *matchReferrals()*

Examples

```r
reaction <- "R-HSA-1369062"
# matchReactionsInPathway(event.id=reaction, type="graph")
# matchReactionsInPathway("R-HSA-5682285", type="row")
```

matchReferrals

MATCH biological referrals

Description

This method retrieves Reactome objects that are connected with the given object in a reverse relationship. For example, to find Pathways containing the given Reaction.

Usage

```r
matchReferrals(
  id = NULL,
  displayName = NULL,
  main = TRUE,
  depth = 1,
  all.depth = FALSE,
  species = NULL,
  type = c("row", "graph")
)
```
multiObjects

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>stId or dbId of a Reactome object</td>
</tr>
<tr>
<td>displayName</td>
<td>displayName of a Reactome object</td>
</tr>
<tr>
<td>main</td>
<td>if set to TRUE, only <strong>first-class</strong> referrals returned</td>
</tr>
<tr>
<td>depth</td>
<td>number of depths</td>
</tr>
<tr>
<td>all.depth</td>
<td>if set to TRUE, connected objects in all depths returned</td>
</tr>
<tr>
<td>species</td>
<td>name or taxon id or dbId or abbreviation of a species</td>
</tr>
<tr>
<td>type</td>
<td>return results as a list of dataframes (<code>row</code>), or as a graph object (<code>graph</code>)</td>
</tr>
</tbody>
</table>

Details

For now it just focuses on biological referrals in the following Classes: "Event", "PhysicalEntity", "Regulation", "CatalystActivity", "ReferenceEntity", "Interaction", "AbstractModifiedResidue".

Value

referrals of the given instance

See Also

Other match: `matchDiseases()`, `matchHierarchy()`, `matchInteractors()`, `matchObject()`, `matchPEroles()`, `matchPaperObjects()`, `matchPrecedingAndFollowingEvents()`, `matchReactionsInPathway()`

Examples

```r
stId <- "R-HSA-112479"
# matchReferrals("R-HSA-112479", main=FALSE, all.depth=TRUE, type="row")
```

multiObjects

Retrieve multiple Reactome objects

Description

The `matchObject` function takes only one id/name at a time, this method allows you to input many ids and get an aggregated table for their detailed information. It can only accept **ids** for now.

Usage

```r
multiObjects(ids, databaseName = "Reactome", speedUp = FALSE, cluster = 2)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ids</td>
<td>Reactome stIds/dbIds, or non-Reactome ids</td>
</tr>
<tr>
<td>databaseName</td>
<td>database name</td>
</tr>
<tr>
<td>speedUp</td>
<td>set TRUE to use doParallel method</td>
</tr>
<tr>
<td>cluster</td>
<td>the number of cluster in makeCluster</td>
</tr>
</tbody>
</table>
Unnest a column of lists in a dataframe

Usage

unnestListCol(df, column = "properties")

Arguments

df data frame where a column to be unnested

column specific column to be unnested

Value

an unnested dataframe for network visualization

Examples

# nodes <- unnestListCol(graph$nodes, "properties")
Index

* internal
  ReactomeGraph4R-package, 2

* match
  matchDiseases, 3
  matchHierarchy, 4
  matchInteractors, 5
  matchObject, 6
  matchPaperObjects, 7
  matchPEroles, 8
  matchPrecedingAndFollowingEvents, 9
  matchReactionsInPathway, 10
  matchReferrals, 11

doParallel, 12

login, 3

makeCluster, 12
matchDiseases, 3, 5, 7–12
matchHierarchy, 4, 4, 5, 7–12
matchInteractors, 4, 5, 5, 7–12
matchObject, 4, 5, 6, 8–13
matchPaperObjects, 4, 5, 7, 7, 9–12
matchPEroles, 4, 5, 7, 8, 8, 10–12
matchPrecedingAndFollowingEvents, 4, 5, 7–9, 9, 11, 12
matchReactionsInPathway, 4, 5, 7–10, 10, 12
matchReferrals, 4, 5, 7–11, 11
multiObjects, 7, 12

ReactomeGraph4R
  (ReactomeGraph4R-package), 2
ReactomeGraph4R-package, 2

unnestListCol, 13