Package ‘epigraphdb’

October 13, 2022

Title  Interface Package for the 'EpiGraphDB' Platform
Version  0.2.3
Description  The interface package to access data from the 'EpiGraphDB' <https://epigraphdb.org> platform.
It provides easy access to the 'EpiGraphDB' platform with functions that query the corresponding REST endpoints on the API <https://api.epigraphdb.org> and return the response data in the 'tibble' data frame format.

URL  https://mrcieu.github.io/epigraphdb-r/

BugReports  https://github.com/MRCIEU/epigraphdb-r/issues

License  GPL-3

Suggests  testthat, roxygen2, knitr, rmarkdown, spelling, devtools,
          usethis, pkgdown, styler, lintr, covr, igraph, gtools, stringr,
          dplyr, ggplot2

Encoding  UTF-8

RoxygenNote  7.1.2

Imports  magrittr, tibble, httr, glue, purrr, jsonlite

VignetteBuilder  knitr

Language  en-GB

NeedsCompilation  no

Author  Yi Liu [cre, aut],
        Valeriu Haberland [aut],
        Marina Vabistsevits [aut],
        Tom Gaunt [aut],
        MRC IEU [cph]

Maintainer  Yi Liu <yi6240.liu@bristol.ac.uk>

Repository  CRAN

Date/Publication  2022-01-14 19:52:45 UTC
R topics documented:

confounder .................................................. 2
cypher .......................................................... 3
drugs_risk_factors ........................................... 4
genetic_cor ..................................................... 5
literature_gwas ............................................... 6
mappings_gene_to_protein .................................... 6
meta_nodes_list ............................................... 7
meta_nodes_list_node .......................................... 8
meta_nodes_search_node ....................................... 9
meta_rels_list ................................................. 10
meta_rels_list_rel ............................................ 11
mr ............................................................. 11
obs_cor .......................................................... 13
ontology_gwas_efo .......................................... 14
pathway ......................................................... 15
pqtl ............................................................. 15
pqtl_list ........................................................ 17
pqtl_pleio ...................................................... 18
protein_in_pathway .......................................... 19
query_epigraphdb ............................................ 19
xqtl_multi.snp_mr ............................................ 21
xqtl_single.snp_mr ........................................... 22

Index .................................................................. 24

confounder ........................................................

MR evidence on confounding traits between exposure and outcome

Description

GET /confounder

Usage

confounder(
   exposure_trait = NULL,
   outcome_trait = NULL,
   type = c("confounder", "intermediate", "reverse_intermediate", "collider"),
   pval_threshold = 1e-05,
   mode = c("table", "raw")
)

Arguments

exposure_trait A trait name, e.g. "Body mass index", leaving exposure_trait as NULL will return MR information related to a specific outcome. **NOTE**: exposure_trait and outcome_trait cannot be both NULL.

outcome_trait A trait name, e.g. "Coronary heart disease", leaving outcome_trait as NULL will return MR information related to a specific exposure_trait. **NOTE**: exposure_trait and outcome_trait cannot be both NULL.

type One in ["confounder", "intermediate", "reverse_intermediate", "collider"] Refer to the confounder view in web application for details

pval_threshold P-value threshold

mode If mode = "table", returns a data frame (a **tibble** as per **tidyverse** convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by **httr**.

Value

Data from GET /confounder

Examples

```r
## Not run:
confounder(exposure_trait = "Body mass index", outcome_trait = "Coronary heart disease")
## End(Not run)
```

---

**cypher**

*Send a query in Cypher to EpiGraphDB*

Description

**NOTE**: this function is intended for advanced uses. Regular users are recommended to use standard query functions

Usage

cypher(query, mode = c("table", "raw"))

Arguments

query A Cypher query.

mode If mode = "table", returns a data frame (a **tibble** as per **tidyverse** convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by **httr**.
drugs_risk_factors

Examples

```r
## Not run:
cypher("MATCH (n:Gwas) RETURN n LIMIT 2")
## End(Not run)
```

---

### drugs_risk_factors

**Drugs for risk factors**

Description

`GET /drugs/risk-factors`

Usage

```r
drugs_risk_factors(trait, pval_threshold = 1e-08, mode = c("table", "raw"))
```

Arguments

- **trait**: A trait name
- **pval_threshold**: P-value threshold
- **mode**: If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by `httr`.

Value

Data from `GET /drugs/risk-factors`

Examples

```r
## Not run:
drugs_risk_factors(trait = "Body mass index")
## End(Not run)
```
**Genetic correlations between traits**

**Description**

GET /genetic-cor

**Usage**

```r
ge genetic_cor(trait, cor_coef_threshold = 0.8, mode = c("table", "raw"))
```

**Arguments**

- **trait**
  - name of the trait, e.g. "body mass index"

- **cor_coef_threshold**
  - correlation coefficient threshold

- **mode**
  - If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

**Value**

Data from GET /genetic_cor

**Examples**

```r
## Not run:
genetic_cor(trait = "Body mass index") %>%
dplyr::glimpse()
## End(Not run)

# Use a different threshold
## Not run:
genetic_cor(trait = "Body mass index", cor_coef_threshold = 0.4) %>%
dplyr::glimpse()
## End(Not run)
```
### literature_gwas

**Literature evidence regarding a GWAS trait**

**Description**

GET /literature/gwas

**Usage**

```
literature_gwas(trait, semmed_predicate = NULL, mode = c("table", "raw"))
```

**Arguments**

- **trait**
  - A trait name

- **semmed_predicate**
  - Either NULL which returns entries from all predicates, or a SemMed predicate e.g. "DIAGNOSES" or "ASSOCIATED_WITH"

- **mode**
  - If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

**Value**

Data from GET /literature/gwas

**Examples**

```r
## Not run:
literature_gwas(trait = "Body mass index")
## End(Not run)
```

### mappings_gene_to_protein

**Return protein uniprot_id from associated genes**

**Description**

POST /mappings/gene-to-protein

**Usage**

```
mappings_gene_to_protein(
  gene_name_list = NULL,
  gene_id_list = NULL,
  by_gene_id = FALSE,
  mode = c("table", "raw")
)
```
Arguments

gene_name_list  List of HGNC symbols of the genes (default)
gene_id_list    List of Ensembl gene IDs (when by_gene_id == TRUE)
by_gene_id      Search for gene ids (Ensembl gene IDs) instead of gene names (HGNC symbols)
mode            If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

Value

Data from POST /mappings/gene-to-protein

Examples

# By HGNC symbols
## Not run:
mappings_gene_to_protein(gene_name_list = c("GCH1", "MYOF"))
## End(Not run)

# By Ensembl Ids
## Not run:
mappings_gene_to_protein(gene_id_list = c("ENSG00000162594", "ENSG00000113302"), by_gene_id = TRUE)
## End(Not run)

---

meta_nodes_list  List meta nodes (e.g. Gwas, Gene, etc.)

Description

GET /meta/nodes/list

Usage

meta_nodes_list(mode = c("raw"))

Arguments

mode            If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

Value

Data from GET /meta/nodes/list
Examples

```r
## Not run:
meta_nodes_list()
## End(Not run)
```

---

**meta_nodes_list_node**  
List nodes under a meta node

**Description**

GET /meta/nodes/{meta_node}/list

**Usage**

```r
meta_nodes_list_node(
  meta_node,
  full_data = TRUE,
  limit = 10,
  offset = 0,
  mode = c("table", "raw")
)
```

**Arguments**

- **meta_node**  
  Name of a meta node (e.g. Gwas). Use `meta_nodes_list` to get the full list of meta nodes.

- **full_data**  
  When False, only return the id and name fields (their specific names differ in specific nodes) for a node. This is useful if you want your queries to return results faster with smaller amount of data requested.

- **limit**  
  Max number of items to retrieve.

- **offset**  
  Number of items to skip. Use `limit` and `offset` in combination to do pagination.

- **mode**  
  If `mode = "table"`, returns a data frame (a **tibble** as per **tidyverse** convention). If `mode = "raw"`, returns a raw response from EpiGraphDB API with minimal parsing done by `httr`.

**Value**

Data from GET /meta/nodes/{meta_node}/list
**Examples**

```r
# List the first 5 Gwas nodes, with only id and name fields
## Not run:
meta_nodes_list_node(meta_node = "Gwas", full_data = FALSE, limit = 5)
## End(Not run)

# List the 6th - 10th Disease nodes, with full properties
## Not run:
meta_nodes_list_node(meta_node = "Disease", full_data = TRUE, limit = 5, offset = 0)
## End(Not run)
```

---

**Description**

**GET /meta/nodes/{meta_node}/search**

**Usage**

```r
meta_nodes_search_node(
  meta_node,
  id = NULL,
  name = NULL,
  limit = 10,
  full_data = TRUE,
  mode = c("table", "raw")
)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>meta_node</td>
<td>Name of a meta node (e.g. Gwas). Use <code>meta_nodes_list</code> to get the full list of meta nodes.</td>
</tr>
<tr>
<td>id</td>
<td>The id field of a node (e.g. &quot;ieu-a-2&quot; for a Gwas). Use EpiGraphDB web UI to get a sense of what those ids are for entities.</td>
</tr>
<tr>
<td>name</td>
<td>The name field of a node (e.g. &quot;body mass index&quot; for a Gwas). Use EpiGraphDB web UI to get a sense of what those names are for entities.</td>
</tr>
<tr>
<td>limit</td>
<td>Max number of items to retrieve.</td>
</tr>
<tr>
<td>full_data</td>
<td>When False, only return the id and name fields (their specific names differ in specific nodes) for a node. This is useful if you want your queries to return results faster with smaller amount of data requested.</td>
</tr>
<tr>
<td>mode</td>
<td>If mode = &quot;table&quot;, returns a data frame (a tibble as per tidyverse convention). If mode = &quot;raw&quot;, returns a raw response from EpiGraphDB API with minimal parsing done by httr.</td>
</tr>
</tbody>
</table>
**meta_rels_list**  

**Value**

Data from GET /meta/nodes/{meta_node}/search

**Examples**

```r
# Search Gwas nodes
## Not run:
meta_nodes_search_node(meta_node = "Gwas", name = "body mass index")

## End(Not run)
```

---

**meta_rels_list**  

*List meta rels (e.g. MR, etc.)*

**Description**

GET /meta/rels/list

**Usage**

```r
meta_rels_list(mode = c("raw"))
```

**Arguments**

- **mode**  
  If mode = "table", returns a data frame (a *tibble* as per *tidyverse* convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by *httr*.

**Value**

Data from GET /meta/rels/list

**Examples**

```r
## Not run:
meta_rels_list()

## End(Not run)
```
meta_rels_list_rel  
List relationships under a meta relationship

Description

GET /meta/rels/{meta_rel}/list

Usage

meta_rels_list_rel(meta_rel, limit = 10, offset = 0, mode = c("table", "raw"))

Arguments

- **meta_rel**
  - Name of a meta relationship (e.g. MR). Use meta_rels_list to get the full list of meta relationships.
- **limit**
  - Max number of items to retrieve.
- **offset**
  - Number of items to skip. Use limit and offset in combination to do pagination.
- **mode**
  - If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

Value

Data from GET /meta/rels/{meta_rel}/list

Examples

# List the first 5 MR relationships
## Not run:
meta_rels_list_rel(meta_rel = "MR_EVE_MR", limit = 5)
## End(Not run)

---

mr  
Return information related to Mendelian Randomisation

Description

GET /mr
Usage

```r
mr(
    exposure_trait = NULL,
    outcome_trait = NULL,
    pval_threshold = 1e-05,
    mode = c("table", "raw")
)
```

Arguments

- `exposure_trait` A trait name, e.g. "Body mass index", leaving exposure_trait as NULL will return MR information related to a specific outcome. **NOTE:** exposure_trait and outcome_trait cannot be both NULL.
- `outcome_trait` A trait name, e.g. "Coronary heart disease", leaving outcome_trait as NULL will return MR information related to a specific exposure_trait. **NOTE:** exposure_trait and outcome_trait cannot be both NULL.
- `pval_threshold` P-value threshold
- `mode` If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

Value

Data from GET /mr

Examples

- # Returns a data frame
  ```r
  ## Not run:
  mr(exposure_trait = "Body mass index", outcome_trait = "Coronary heart disease")
  ## End(Not run)
  ```

- # Returns raw response
  ```r
  ## Not run:
  mr(
      exposure_trait = "Body mass index", outcome_trait = "Coronary heart disease",
      mode = "raw"
  ) %>% str()
  ## End(Not run)
  ```

- # Use a different threshold
  ```r
  ## Not run:
  mr(exposure_trait = "Body mass index", pval_threshold = 1e-8)
  ## End(Not run)
  ```
Description

GET /obs-cor

Usage

obs_cor(trait, cor_coef_threshold = 0.8, mode = c("table", "raw"))

Arguments

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trait</td>
<td>name of the trait, e.g. &quot;body mass index&quot;</td>
</tr>
<tr>
<td>cor_coef_threshold</td>
<td>correlation coefficient threshold</td>
</tr>
<tr>
<td>mode</td>
<td>If mode = &quot;table&quot;, returns a data frame (a tibble as per tidyverse convention). If mode = &quot;raw&quot;, returns a raw response from EpiGraphDB API with minimal parsing done by httr.</td>
</tr>
</tbody>
</table>

Value

Data from GET /obs-cor

Examples

```r
## Not run:
obs_cor(trait = "Body mass index (BMI)") %>%
dplyr::glimpse()
## End(Not run)

# Use a different threshold
## Not run:
obs_cor(trait = "Body mass index (BMI)", cor_coef_threshold = 0.8) %>%
dplyr::glimpse()
## End(Not run)
```
ontology_gwas_efo

Ontology association between EFO term and Gwas

Description

GET /ontology/gwas-efo

Usage

ontology_gwas_efo(
  trait = NULL,
  efo_term = NULL,
  fuzzy = TRUE,
  mode = c("table", "raw")
)

Arguments

trait trait name, e.g. "body mass"
efo_term EFO term, e.g. "systolic blood pressure"
fuzzy whether query with exact matching (FALSE) or fuzzy matching (default, TRUE)
mode If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

Value

Data from GET /ontology/gwas-efo

Examples

## Not run:
ontology_gwas_efo(trait = "blood", fuzzy = FALSE)

## End(Not run)

## Not run:
ontology_gwas_efo(efo_term = "blood pressure", fuzzy = FALSE)

## End(Not run)
pathway

Description

GET /pathway

Usage

pathway(trait, pval_threshold = 1e-05, mode = c("table", "raw"))

Arguments

trait A trait name
pval_threshold P-value threshold
mode If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

Value

Data from GET /pathway

Examples

## Not run:
pathway(trait = "Body mass index")

## End(Not run)

pqtl

Return information related to the pQTL analysis

Description

GET /pqtl/

Usage

pqtl(
query,
rtype = c("mrres", "simple", "sglmar", "inst", "sense"),
pvalue = 0.05,
searchflag = c("traits", "proteins"),
mode = c("table", "raw")
)
Arguments

query  (Required) A protein coding gene name or a trait name, e.g. "ADAM19" or "Inflammatory bowel disease" which cannot be NULL.

rtype  (Optional) A type of data to be extracted, which can be one of these options:
1. simple: Basic summary
2. mrres: MR results (DEFAULT)
3. sglmr: Single SNP MR results
4. inst: SNP information
5. sense: Sensitivity analysis NOTE: mrres is a DEFAULT option.

pvalue  (Optional) A pvalue threshold for MR results with the DEFAULT set to 0.05. NOTE: this threshold applies to any rtype chosen.

searchflag  (Required) A flag to indicate whether you are searching for proteins or traits which cannot be NULL. If query is a protein name, then this flag should be "proteins"; if query is a trait, this flag should be "traits". NOTE: if the wrong flag is chosen for query, there will be no result returned.

mode  If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

Value

Data from GET /pqtl/

Examples

# Returns a data frame of MR results, while searching for proteins
## Not run:
pqtl(query = "ADAM19", searchflag = "proteins")

## End(Not run)

# Returns a data frame with SNP information, while searching for traits
## Not run:
pqtl(
    query = "Inflammatory bowel disease",
    rtype = "inst",
    searchflag = "traits"
)

## End(Not run)

# Change a pvalue threshold (the default is 0.05)
## Not run:
pqtl(
    query = "Inflammatory bowel disease",
    rtype = "inst",
    pvalue = 1.0,
    searchflag = "traits"
pqtl_list

Return a list of all proteins/exposures or traits/outcomes available in the database

Description

GET /pqtl/list/

Usage

pqtl_list(flag = c("exposures", "outcomes"), mode = c("table", "raw"))

Arguments

flag  (Optional) A flag which indicates whether the list of exposures (if "exposures") or outcomes (if "outcomes") should be returned. The DEFAULT is "exposures".

mode  If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

Value

Data from GET /pqtl/list/

Examples

# Returns a list of available proteins (exposures)
## Not run:
pqtl_list()

## End(Not run)

# Returns a list of available traits (outcomes)
## Not run:
pqtl_list(flag = "outcomes")

## End(Not run)
pqtl_pleio  

Return information related to the pleiotropy of SNPs

Description

GET /pqtl/pleio/

Usage

pqtl_pleio(
  rsid = NULL,
  prflag = c("proteins", "count"),
  mode = c("table", "raw")
)

Arguments

rsid  
(Required) A SNP identified by rsID which cannot be NULL.

prflag  
(Optional) A flag which determines whether the number (if "count") or names (if "proteins") of the associated proteins should be returned. The DEFAULT value is "proteins".

mode  
(Optional) If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

Value

Data from GET /pqtl/pleio/

Examples

# Returns a data frame of associated proteins
## Not run:
pqtl_pleio(rsid = "rs1260326")
## End(Not run)

# Returns a number of associated proteins
## Not run:
pqtl_pleio(rsid = "rs1260326", prflag = "count")
## End(Not run)
**protein_in_pathway**

*For the list of proteins, returns their associated pathway data*

### Description

**POST /protein/in-pathway**

### Usage

```r
generate_pathway_data(uniprot_id_list, mode = c("table", "raw"))
```

### Arguments

- **uniprot_id_list**
  
  A list of protein UniProt IDs

- **mode**
  
  If `mode = "table"`, returns a data frame (a **tibble** as per **tidyverse** convention). If `mode = "raw"`, returns a raw response from EpiGraphDB API with minimal parsing done by **httr**.

### Value

Data from **POST /protein/in-pathway**

### Examples

```r
## Not run:
generate_pathway_data(uniprot_id_list = c("014933", "060674", "P32455"))
## End(Not run)
```

---

**query_epigraphdb**

*Send data request to an EpiGraphDB API endpoint*

### Description

This is a general purpose function to send data request which can be used when there has not been an R equivalent package function to an API endpoint. Underneath this is a wrapper around **httr** functions with better handling of returned status.
Usage

query_epigraphdb(
  route,
  params = NULL,
  mode = c("raw", "table"),
  method = c("GET", "POST"),
  retry_times = 3,
  retry_pause_min = 1
)

Arguments

route An EpiGraphDB API endpoint route, e.g. "/mr" or "/confounder". Consult the EpiGraphDB API documentation.

params A list of parameters associated with the query endpoint.

mode c("raw", "table"), if "table" then the query handler will try to convert the returned data to a tibble dataframe. NOTE: The default mode is "raw" which will NOT convert the returned response to a dataframe. This is different to functions that query topic endpoints which default to return a dataframe. Explicitly specify mode = "table" when needed.

method Type of HTTP (GET, POST, PUT, etc.) method.

NOTE: When sending a POST request where a specific parameter is specified as a list on the API, and if the equivalent in R is a vector of length 1, you should wrap this parameter in I(), e.g. I(c("APOE")) to avoid auto unboxing. For details, please refer to 

retry_times Number of times the function will retry the request to the API.

retry_pause_min Minimum number of seconds to wait for the next retry.

Value

Data from an EpiGraphDB API endpoint.

Examples

# GET /mr
# equivalent to `mr(exposure_trait = "Body mass index", outcome_trait = "Coronary heart disease")`
## Not run:
query_epigraphdb(
  route = "/mr",
  params = list(
    exposure_trait = "Body mass index",
    outcome_trait = "Coronary heart disease"
  ),
  mode = "table"
)

## End(Not run)
# GET /meta/nodes/Gwas/list
## Not run:
query_epigraphdb(
  route = "/meta/nodes/Gwas/list",
  params = list(
    limit = 5,
    offset = 0
  )
)
## End(Not run)

# POST /protein/ppi
## Not run:
query_epigraphdb(
  route = "/protein/ppi",
  params = list(
    uniprot_id_list = c("P30793", "Q9NZM1", "O95236"),
    method = "POST"
  )
)
## End(Not run)

# error handling
## Not run:
tryCatch(
  query_epigraphdb(
    route = "/mr",
    params = list(
      exposure_trait = NULL,
      outcome_trait = NULL
    ),
    retry_times = 0
  ),
  error = function(e) {
    message(e)
  }
)
## End(Not run)

---

**xqtl_multi_snpr**

*Multi SNP QTL MR evidence*

**Description**

GET /xqtl/multi-snp-mr
Usage

xqtl_multi_snp_mr(
    exposure_gene = NULL,
    outcome_trait = NULL,
    mr_method = c("IVW", "Egger"),
    qtl_type = c("eQTL", "pQTL"),
    pval_threshold = 1e-05,
    mode = c("table", "raw")
)

Arguments

exposure_gene  Name of the exposure gene
outcome_trait  Name of the outcome trait
mr_method      "IVW" or "Egger"
qtl_type       "eQTL" or "pQTL"
pval_threshold P-value threshold
mode           If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

Value

Data from GET /xqtl/multi-snp-mr

Examples

## Not run:
 xqtl_multi_snp_mr(outcome_trait = "Coronary heart disease")

## End(Not run)

xqtl_single_snp_mr  Single SNP QTL MR evidence

Description

GET /xqtl/single-snp-mr

Usage

xqtl_single_snp_mr(
    exposure_gene = NULL,
    outcome_trait = NULL,
    snp = NULL,
    qtl_type = c("eQTL", "pQTL"),
)
xqtl_single_snp_mr

    pval_threshold = 1e-05,
    mode = c("table", "raw")
)

Arguments

  exposure_gene  Name of the exposure gene
  outcome_trait  Name of the outcome trait
  snp            SNP rsid
  qtl_type       "eQTL" or "pQTL"
  pval_threshold P-value threshold
  mode           If mode = "table", returns a data frame (a tibble as per tidyverse convention). If mode = "raw", returns a raw response from EpiGraphDB API with minimal parsing done by httr.

Value

  Data from GEET /xqtl/single-snp-mr

Examples

  ## Not run:
  xqtl_single_snp_mr(outcome_trait = "Coronary heart disease")

  ## End(Not run)
Index

confounder, 2
cypher, 3
drugs_risk_factors, 4

genetic_cor, 5

literature_gwas, 6
mappings_gene_to_protein, 6
meta_nodes_list, 7
meta_nodes_list_node, 8
meta_nodes_search_node, 9
meta_rels_list, 10
meta_rels_list_rel, 11
mr, 11

obs_cor, 13
ontology_gwas_efo, 14

pathway, 15
pqtl, 15
pqtl_list, 17
pqtl_pleio, 18
protein_in_pathway, 19

query_epigraphdb, 19

xqtl_multi_snp_mr, 21
xqtl_single_snp_mr, 22