

# Package ‘otargen’

July 22, 2025

**Type** Package

**Title** Access Open Target

**Version** 2.0.0

**Date** 2025-07-08

**Maintainer** Amir Feizi <afeizi@gmail.com>

## Description

Interact seamlessly with Open Target GraphQL endpoint to query and retrieve tidy data tables, facilitating the analysis of gene, disease, drug, and genetic data. For more information about the Open Target API (<<https://platform.opentargets.org/api>>).

**Depends** R (>= 3.1.0)

**Imports** ghql, cli, dplyr, jsonlite, magrittr, tibble, tidyr, http

**License** MIT + file LICENSE

**Encoding** UTF-8

**Suggests** testthat (>= 3.0.0)

**Config/testthat/edition** 3

**RoxygenNote** 7.3.2

**URL** <https://amirfeizi.github.io/otargen/>

**BugReports** <https://github.com/amirfeizi/otargen/issues>

**NeedsCompilation** no

**Author** Amir Feizi [aut, cre]

**Repository** CRAN

**Date/Publication** 2025-07-15 19:10:02 UTC

## Contents

|                              |   |
|------------------------------|---|
| adverseEventsQuery . . . . . | 2 |
| chemblQuery . . . . .        | 3 |
| clinVarQuery . . . . .       | 4 |
| compGenomicsQuery . . . . .  | 5 |

|                                        |           |
|----------------------------------------|-----------|
| depMapQuery . . . . .                  | 5         |
| europaPMCQuery . . . . .               | 6         |
| geneBurdenQuery . . . . .              | 7         |
| geneOntologyQuery . . . . .            | 7         |
| geneticConstraintQuery . . . . .       | 8         |
| genomicsEnglandQuery . . . . .         | 9         |
| gwasColocalisation . . . . .           | 9         |
| gwasCredibleSet . . . . .              | 11        |
| gwasCredibleSetsQuery . . . . .        | 12        |
| hallmarksQuery . . . . .               | 12        |
| indicationsQuery . . . . .             | 13        |
| interactionsQuery . . . . .            | 14        |
| knownDrugsChEMBLQuery . . . . .        | 14        |
| knownDrugsGeneQuery . . . . .          | 15        |
| locus2GeneQuery . . . . .              | 16        |
| mechanismsOfActionQuery . . . . .      | 16        |
| mousePhenotypesQuery . . . . .         | 17        |
| orphanetQuery . . . . .                | 18        |
| overlapInfoForStudy . . . . .          | 18        |
| pathwaysQuery . . . . .                | 19        |
| pharmacogenomicsChEMBLQuery . . . . .  | 20        |
| pharmacogenomicsGeneQuery . . . . .    | 21        |
| pharmacogenomicsVariantQuery . . . . . | 21        |
| qtlCredibleSetsQuery . . . . .         | 22        |
| safetyQuery . . . . .                  | 23        |
| sharedTraitStudiesQuery . . . . .      | 23        |
| uniprotLiteratureQuery . . . . .       | 24        |
| uniProtVariantsQuery . . . . .         | 25        |
| variantEffectPredictorQuery . . . . .  | 25        |
| variantEffectQuery . . . . .           | 26        |
| variantsQuery . . . . .                | 26        |
| <b>Index</b>                           | <b>28</b> |

---

|                    |                                                           |
|--------------------|-----------------------------------------------------------|
| adverseEventsQuery | <i>Retrieve Adverse Events data for a specified drug.</i> |
|--------------------|-----------------------------------------------------------|

---

## Description

This function queries the Open Targets GraphQL API to retrieve adverse events data for a specified drug.

## Usage

```
adverseEventsQuery(chemblId, index = 0, size = 10)
```

**Arguments**

|          |                                                               |
|----------|---------------------------------------------------------------|
| chemblId | Character: ChEMBL ID of the target drug (e.g., "CHEMBL1016"). |
| index    | Integer: Page index for pagination (default: 0).              |
| size     | Integer: Number of records to retrieve (default: 10).         |

**Value**

Returns a tibble containing adverse events data for the specified drug.

**Examples**

```
## Not run:
result <- adverseEventsQuery(chemblId = "CHEMBL1016", size = 10)
result <- adverseEventsQuery(chemblId = "CHEMBL1016", index = 0, size = 10)

## End(Not run)
```

---

|             |                                                               |
|-------------|---------------------------------------------------------------|
| chemblQuery | <i>Retrieve ChEMBL data for a specified gene and disease.</i> |
|-------------|---------------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve ChEMBL evidence data for a specified gene and disease.

**Usage**

```
chemblQuery(ensemblId, efoId, cursor = NULL, size = 10)
```

**Arguments**

|           |                                                                     |
|-----------|---------------------------------------------------------------------|
| ensemblId | Character: ENSEMBL ID of the target gene (e.g., "ENSG00000080815"). |
| efoId     | Character: EFO ID of the target disease (e.g., "MONDO_0004975").    |
| cursor    | Character: Cursor for pagination (default: NULL).                   |
| size      | Integer: Number of records to retrieve (default: 10).               |

**Value**

Returns a tibble containing ChEMBL evidence data for the specified gene and disease.

**Examples**

```
## Not run:
result <- chemblQuery(ensemblId = "ENSG00000080815", efoId =
"MONDO_0004975",
  size = 10)
result <- chemblQuery(ensemblId = "ENSG00000080815", efoId =
"MONDO_0004975",
  cursor = NULL, size = 10)

## End(Not run)
```

---

clinVarQuery

*Retrieve ClinVar data for a specified gene and disease.*


---

**Description**

This function queries the Open Targets GraphQL API to retrieve ClinVar evidence data for a specified gene and disease.

**Usage**

```
clinVarQuery(ensemblId, efoId, cursor = NULL, size = 10)
```

**Arguments**

|           |                                                                     |
|-----------|---------------------------------------------------------------------|
| ensemblId | Character: ENSEMBL ID of the target gene (e.g., "ENSG00000080815"). |
| efoId     | Character: EFO ID of the target disease (e.g., "MONDO_0004975").    |
| cursor    | Character: Cursor for pagination (default: NULL).                   |
| size      | Integer: Number of records to retrieve (default: 10).               |

**Value**

Returns a tibble containing ClinVar evidence data for the specified gene and disease.

**Examples**

```
## Not run:
result <- clinVarQuery(ensemblId = "ENSG00000080815", efoId =
"MONDO_0004975", size = 10)
result <- clinVarQuery(ensemblId = "ENSG00000080815", efoId =
"MONDO_0004975", cursor = NULL, size = 10)

## End(Not run)
```

---

|                   |                                                                 |
|-------------------|-----------------------------------------------------------------|
| compGenomicsQuery | <i>Retrieve Comparative Genomics data for a specified gene.</i> |
|-------------------|-----------------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve comparative genomics data for a specified gene.

**Usage**

```
compGenomicsQuery(ensemblId)
```

**Arguments**

ensemblId      Character: ENSEMBL ID of the target gene (e.g., ENSG00000169174).

**Value**

Returns a data frame containing comparative genomics data for the specified gene.

**Examples**

```
## Not run:  
result <- compGenomicsQuery(ensemblId = "ENSG00000169174")  
  
## End(Not run)
```

---

|             |                                                                |
|-------------|----------------------------------------------------------------|
| depMapQuery | <i>Retrieve DepMap Essentiality data for a specified gene.</i> |
|-------------|----------------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve DepMap essentiality data for a specified gene.

**Usage**

```
depMapQuery(ensgId)
```

**Arguments**

ensgId      Character: ENSEMBL ID of the target gene (e.g., ENSG00000141510).

**Value**

Returns a tibble containing DepMap essentiality data for the specified gene.

**Examples**

```
## Not run:
result <- depMapQuery(ensgId = "ENSG00000141510")

## End(Not run)
```

---

|                |                                                                   |
|----------------|-------------------------------------------------------------------|
| europePMCQuery | <i>Retrieve Europe PMC data for a specified gene and disease.</i> |
|----------------|-------------------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve Europe PMC evidence data for a specified gene and disease.

**Usage**

```
europePMCQuery(ensemblId, efoId, cursor = NULL, size = 50)
```

**Arguments**

|           |                                                                     |
|-----------|---------------------------------------------------------------------|
| ensemblId | Character: ENSEMBL ID of the target gene (e.g., "ENSG00000080815"). |
| efoId     | Character: EFO ID of the target disease (e.g., "MONDO_0004975").    |
| cursor    | Character: Cursor for pagination (default: NULL).                   |
| size      | Integer: Number of records to retrieve (default: 50).               |

**Value**

Returns a tibble containing Europe PMC evidence data for the specified gene and disease.

**Examples**

```
## Not run:
result <- europePMCQuery(ensemblId = "ENSG00000080815",
  efoId = "MONDO_0004975", size = 50)
result <- europePMCQuery(ensemblId = "ENSG00000080815",
  efoId = "MONDO_0004975", cursor = NULL, size = 50)

## End(Not run)
```

---

|                 |                                                                    |
|-----------------|--------------------------------------------------------------------|
| geneBurdenQuery | <i>Retrieve Gene Burden data for a specified gene and disease.</i> |
|-----------------|--------------------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve gene burden evidence data for a specified gene and disease.

**Usage**

```
geneBurdenQuery(ensemblId, efoId, size = 3500)
```

**Arguments**

|           |                                                                     |
|-----------|---------------------------------------------------------------------|
| ensemblId | Character: ENSEMBL ID of the target gene (e.g., "ENSG00000137642"). |
| efoId     | Character: EFO ID of the target disease (e.g., "MONDO_0004975").    |
| size      | Integer: Number of records to retrieve (default: 3500).             |

**Value**

Returns a tibble containing gene burden evidence data for the specified gene and disease.

**Examples**

```
## Not run:  
result <- geneBurdenQuery(ensemblId = "ENSG00000137642", efoId =  
"MONDO_0004975", size = 3500)  
  
## End(Not run)
```

---

|                   |                                                          |
|-------------------|----------------------------------------------------------|
| geneOntologyQuery | <i>Retrieve Gene Ontology data for a specified gene.</i> |
|-------------------|----------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve gene ontology data for a specified gene.

**Usage**

```
geneOntologyQuery(ensgId)
```

**Arguments**

|        |                                                                   |
|--------|-------------------------------------------------------------------|
| ensgId | Character: ENSEMBL ID of the target gene (e.g., ENSG00000141510). |
|--------|-------------------------------------------------------------------|

**Value**

Returns a tibble containing gene ontology data for the specified gene.

**Examples**

```
## Not run:  
result <- geneOntologyQuery(ensgId = "ENSG00000141510")  
  
## End(Not run)
```

---

geneticConstraintQuery

*Retrieve Genetic Constraint data for a specified gene.*

---

**Description**

This function queries the Open Targets Platform GraphQL API to retrieve genetic constraint data for a specified gene, such as pLI or LOEUF scores.

**Usage**

```
geneticConstraintQuery(ensgId)
```

**Arguments**

ensgId                      Character: ENSEMBL ID of the target gene (e.g., "ENSG00000141510").

**Value**

Returns a tibble containing genetic constraint data for the specified gene.

**Examples**

```
## Not run:  
result <- geneticConstraintQuery(ensgId = "ENSG00000141510")  
  
## End(Not run)
```



---

genomicsEnglandQuery     *Retrieve Genomics England data for a specified gene and disease.*

---

### Description

This function queries the Open Targets GraphQL API to retrieve Genomics England evidence data for a specified gene and disease.

### Usage

```
genomicsEnglandQuery(ensemblId, efoId, size = 3500)
```

### Arguments

|           |                                                                     |
|-----------|---------------------------------------------------------------------|
| ensemblId | Character: ENSEMBL ID of the target gene (e.g., "ENSG00000080815"). |
| efoId     | Character: EFO ID of the target disease (e.g., "MONDO_0004975").    |
| size      | Integer: Number of records to retrieve (default: 3500).             |

### Value

Returns a tibble containing Genomics England evidence data for the specified gene and disease.

### Examples

```
## Not run:
result <- genomicsEnglandQuery(ensemblId = "ENSG00000080815", efoId =
"MONDO_0004975", size = 3500)

## End(Not run)
```

---

gwasColocalisation     *Retrieve calculated GWAS colocalisation data*

---

### Description

This function retrieves colocalisation data for a specific study locus from a GWAS study with other GWAS studies. It returns a data frame of the studies that colocalise with the input study locus, including details on the study, reported trait, index variant, and calculated colocalisation method outputs.

### Usage

```
gwasColocalisation(study_locus_id, size = 500, index = 0)
```

## Arguments

`study_locus_id` Character: Open Target Genetics generated ID for the study locus (e.g., "5a86bfd40d2ebecf6ce97bbe8a737512").

`size` Integer: Number of rows to fetch per page. Default: 500.

`index` Integer: Page index for pagination. Default: 0.

## Value

Returns a data frame of the studies that colocalise with the input study locus. The table consists of the following data structure:

- `study.studyId`: *Character vector*. Study identifier.
- `study.traitReported`: *Character vector*. Reported trait associated with the colocalisation.
- `study.projectId`: *Character vector*. Project identifier for the study.
- `study.publicationFirstAuthor`: *Character vector*. First author of the publication.
- `indexVariant.id`: *Character vector*. Index variant identifier.
- `indexVariant.position`: *Integer vector*. Index variant position.
- `indexVariant.chromosome`: *Character vector*. Index variant chromosome.
- `indexVariant.referenceAllele`: *Character vector*. Reference allele of the variant.
- `indexVariant.alternateAllele`: *Character vector*. Alternate allele of the variant.
- `pValueMantissa`: *Numeric vector*. Mantissa of the p-value for the colocalisation.
- `pValueExponent`: *Integer vector*. Exponent of the p-value for the colocalisation.
- `numberColocalisingVariants`: *Integer vector*. Number of colocalising variants.
- `colocalisationMethod`: *Character vector*. Method used for colocalisation analysis.
- `h3`: *Numeric vector*. H3 value associated with the colocalisation.
- `h4`: *Numeric vector*. H4 value associated with the colocalisation.
- `clpp`: *Numeric vector*. Colocalisation posterior probability.
- `betaRatioSignAverage`: *Numeric vector*. Average sign of the beta ratio.

## References

Giambartolomei, Claudia et al. "Bayesian test for colocalisation between pairs of genetic association studies using summary statistics." *PLoS genetics* vol. 10,5 e1004383. 15 May. 2014, doi:10.1371/journal.pgen.1004383

## Examples

```
## Not run:
colocalisation_data <- gwasColocalisation(study_locus_id = "5a86bfd40d2ebecf6ce97bbe8a737512",
  size = 500, index = 0)

## End(Not run)
```

---

|                 |                                 |
|-----------------|---------------------------------|
| gwasCredibleSet | Retrieve GWAS credible set data |
|-----------------|---------------------------------|

---

## Description

Provided with a study ID and a lead variant ID, this function returns a data frame consisting of all the associated credible set tag variants with the corresponding statistical data.

## Usage

```
gwasCredibleSet(study_id, variant_id)
```

## Arguments

|            |                                                                                                  |
|------------|--------------------------------------------------------------------------------------------------|
| study_id   | Character: Study ID(s) generated by Open Targets (e.g GCST90002357).                             |
| variant_id | Character: generated ID for variants by Open Targets (e.g. 1_154119580_C_A) or rsId (rs2494663). |

## Value

Returns a data frame of results from the credible set of variants for a specific lead variant with the following columns:

- tagVariant.id: *Data frame*. A table of IDs of the tag variant.
- tagVariant.rsId: *Character vector*. rsID of the tag variant.
- beta: *Numeric*. Beta value.
- postProb: *Numeric*. Posterior probability.
- pval: *Numeric*. P-value.
- se: *Numeric*. Standard error.
- MultisignalMethod: *Character vector*. Multisignal method.
- logABF: *Numeric*. Logarithm of approximate Bayes factor.
- is95: *Logical*. Indicates if the variant has a 95
- is99: *Logical*. Indicates if the variant has a 99

## Examples

```
## Not run:  
result <- gwasCredibleSet(study_id="GCST90002357",  
  variant_id="1_154119580_C_A")  
result <- gwasCredibleSet(study_id="GCST90002357", variant_id="rs2494663")  
  
## End(Not run)
```

---

gwasCredibleSetsQuery    *Retrieve GWAS Credible Sets data for a specified target and disease.*

---

### Description

This function queries the Open Targets Platform GraphQL API to retrieve GWAS credible sets evidence data for a specified target gene and disease.

### Usage

```
gwasCredibleSetsQuery(ensemblId, efoId, size = 500)
```

### Arguments

|           |                                                      |
|-----------|------------------------------------------------------|
| ensemblId | Character. Ensembl gene ID, e.g., "ENSG00000169174". |
| efoId     | Character. EFO disease ID, e.g., "EFO_0004911".      |
| size      | Integer. Number of rows to fetch. Default: 500.      |

### Value

A tibble with credible set evidence or NULL if no data found.

### Examples

```
## Not run:
result <- gwasCredibleSetsQuery(
  ensemblId = "ENSG00000169174",
  efoId = "EFO_0004911",
  size = 5
)
print(result)

## End(Not run)
```

---

hallmarksQuery    *Retrieve Cancer Hallmarks data for a specified gene.*

---

### Description

This function queries the Open Targets GraphQL API to retrieve cancer hallmarks data for a specified gene.

### Usage

```
hallmarksQuery(ensgId)
```

**Arguments**

ensgId                      Character: ENSEMBL ID of the target gene (e.g., ENSG00000141510).

**Value**

Returns a tibble containing cancer hallmarks data for the specified gene.

**Examples**

```
## Not run:
result <- hallmarksQuery(ensgId = "ENSG00000141510")

## End(Not run)
```

---

|                  |                                                        |
|------------------|--------------------------------------------------------|
| indicationsQuery | <i>Retrieve Indications data for a specified drug.</i> |
|------------------|--------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve indications data for a specified drug.

**Usage**

```
indicationsQuery(chemblId)
```

**Arguments**

chemblId                      Character: ChEMBL ID of the target drug (e.g., "CHEMBL1016").

**Value**

Returns a tibble containing indications data for the specified drug.

**Examples**

```
## Not run:
result <- indicationsQuery(chemblId = "CHEMBL1016")

## End(Not run)
```

---

|                   |                                                         |
|-------------------|---------------------------------------------------------|
| interactionsQuery | <i>Retrieve Interactions data for a specified gene.</i> |
|-------------------|---------------------------------------------------------|

---

### Description

This function queries the Open Targets GraphQL API to retrieve molecular interaction data for a specified gene.

### Usage

```
interactionsQuery(ensgId, sourceDatabase = NULL, index = 0, size = 10)
```

### Arguments

|                |                                                                               |
|----------------|-------------------------------------------------------------------------------|
| ensgId         | Character: ENSEMBL ID of the target gene (e.g., ENSG00000141510).             |
| sourceDatabase | Character: Source database for interactions (e.g., "intact") (default: NULL). |
| index          | Integer: Page index for pagination (default: 0).                              |
| size           | Integer: Number of records to retrieve (default: 10).                         |

### Value

Returns a tibble containing interactions data for the specified gene.

### Examples

```
## Not run:
result <- interactionsQuery(ensgId = "ENSG00000141510",
  sourceDatabase = "intact", size = 10)
result <- interactionsQuery(ensgId = "ENSG00000141510",
  sourceDatabase = "intact", index = 0, size = 10)

## End(Not run)
```

---

|                       |                                                        |
|-----------------------|--------------------------------------------------------|
| knownDrugsChEMBLQuery | <i>Retrieve Known Drugs data for a specified drug.</i> |
|-----------------------|--------------------------------------------------------|

---

### Description

This function queries the Open Targets GraphQL API to retrieve known drugs data for a specified drug.

### Usage

```
knownDrugsChEMBLQuery(chemblId, cursor = NULL, freeTextQuery = NULL, size = 10)
```

**Arguments**

|               |                                                               |
|---------------|---------------------------------------------------------------|
| chemblId      | Character: ChEMBL ID of the target drug (e.g., "ChEMBL1016"). |
| cursor        | Character: Cursor for pagination (default: NULL).             |
| freeTextQuery | Character: Free text query to filter results (default: NULL). |
| size          | Integer: Number of records to retrieve (default: 10).         |

**Value**

Returns a tibble containing known drugs data for the specified drug.

**Examples**

```
## Not run:
result <- knownDrugsChemblQuery(chemblId = "ChEMBL1016", size = 10)
result <- knownDrugsChemblQuery(chemblId = "ChEMBL1016", cursor = NULL,
freeTextQuery = NULL, size = 10)

## End(Not run)
```

---

|                     |                                                        |
|---------------------|--------------------------------------------------------|
| knownDrugsGeneQuery | <i>Retrieve Known Drugs data for a specified gene.</i> |
|---------------------|--------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve known drugs data for a specified gene.

**Usage**

```
knownDrugsGeneQuery(ensgId, cursor = NULL, freeTextQuery = NULL, size = 10)
```

**Arguments**

|               |                                                                   |
|---------------|-------------------------------------------------------------------|
| ensgId        | Character: ENSEMBL ID of the target gene (e.g., ENSG00000169174). |
| cursor        | Character: Cursor for pagination (default: NULL).                 |
| freeTextQuery | Character: Free text query to filter results (default: NULL).     |
| size          | Integer: Number of records to retrieve (default: 10).             |

**Value**

Returns a data frame containing known drugs data for the specified gene.

**Examples**

```
## Not run:
result <- knownDrugsGeneQuery(ensgId = "ENSG00000169174", size = 10)
result <- knownDrugsGeneQuery(ensgId = "ENSG00000169174",
  cursor = NULL, freeTextQuery = NULL, size = 10)

## End(Not run)
```

---

|                 |                                                                             |
|-----------------|-----------------------------------------------------------------------------|
| locus2GeneQuery | <i>Retrieve Locus-to-Gene Predictions data for a specified study locus.</i> |
|-----------------|-----------------------------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve locus-to-gene prediction data for a specified study locus.

**Usage**

```
locus2GeneQuery(studyLocusId)
```

**Arguments**

studyLocusId     Character: ID of the target study locus (e.g., "fa375739ca2a6b825ce5cc69d117e84b").

**Value**

Returns a tibble containing locus-to-gene prediction data for the specified study locus.

**Examples**

```
## Not run:
result <- locus2GeneQuery(studyLocusId = "fa375739ca2a6b825ce5cc69d117e84b")

## End(Not run)
```

---

|                         |                                                                 |
|-------------------------|-----------------------------------------------------------------|
| mechanismsOfActionQuery | <i>Retrieve Mechanisms of Action data for a specified drug.</i> |
|-------------------------|-----------------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve mechanisms of action data for a specified drug.

**Usage**

```
mechanismsOfActionQuery(chemblId)
```



**Arguments**

chemblId            Character: ChEMBL ID of the target drug (e.g., "CHEMBL1016").

**Value**

Returns a tibble containing mechanisms of action data for the specified drug.

**Examples**

```
## Not run:
result <- mechanismsOfActionQuery(chemblId = "CHEMBL1016")

## End(Not run)
```

---

mousePhenotypesQuery    *Retrieve Mouse Phenotypes data for a specified gene.*

---

**Description**

This function queries the Open Targets GraphQL API to retrieve mouse phenotypes data for a specified gene.

**Usage**

```
mousePhenotypesQuery(ensemblId)
```

**Arguments**

ensemblId            Character: ENSEMBL ID of the target gene (e.g., ENSG00000169174).

**Value**

Returns a data frame containing mouse phenotypes data for the specified gene.

**Examples**

```
## Not run:
result <- mousePhenotypesQuery(ensemblId = "ENSG00000169174")

## End(Not run)
```

---

|               |                                                                 |
|---------------|-----------------------------------------------------------------|
| orphanetQuery | <i>Retrieve Orphanet data for a specified gene and disease.</i> |
|---------------|-----------------------------------------------------------------|

---

### Description

This function queries the Open Targets GraphQL API to retrieve Orphanet evidence data for a specified gene and disease.

### Usage

```
orphanetQuery(ensemblId, efoId, size = 3500)
```

### Arguments

|           |                                                                     |
|-----------|---------------------------------------------------------------------|
| ensemblId | Character: ENSEMBL ID of the target gene (e.g., "ENSG00000080815"). |
| efoId     | Character: EFO ID of the target disease (e.g., "MONDO_0004975").    |
| size      | Integer: Number of records to retrieve (default: 3500).             |

### Value

Returns a tibble containing Orphanet evidence data for the specified gene and disease.

### Examples

```
## Not run:
result <- orphanetQuery(ensemblId = "ENSG00000080815", efoId =
"MONDO_0004975", size = 3500)

## End(Not run)
```

---

|                     |                                                                 |
|---------------------|-----------------------------------------------------------------|
| overlapInfoForStudy | <i>Retrieves overlap info for a study and a list of studies</i> |
|---------------------|-----------------------------------------------------------------|

---

### Description

For an input study ID and a list of other study IDs, this function returns two elements. One contains the overlap information in a table format, and the other element is the variant intersection set, representing an overlap between two variants of the two given studies.

### Usage

```
overlapInfoForStudy(study_id, study_ids = list())
```

**Arguments**

|           |                                                                                                  |
|-----------|--------------------------------------------------------------------------------------------------|
| study_id  | Character: Study ID(s) generated by Open Targets (e.g GCST90002357).                             |
| study_ids | Character: generated ID for variants by Open Targets (e.g. 1_154119580_C_A) or rsId (rs2494663). |

**Value**

A list containing a data frame of overlap information and the variant intersection set. The overlap information table (overlap\_info) consists of the following columns:

- studyId: *Character vector*. Study ID.
- traitReported: *Character vector*. Reported trait.
- traitCategory: *Character vector*. Trait category.
- variantIdA: *Character vector*. Variant ID from study A.
- variantIdB: *Character vector*. Variant ID from study B.
- overlapAB: *Integer vector*. Number of overlaps between variants A and B.
- distinctA: *Integer vector*. Number of distinct variants in study A.
- distinctB: *Integer vector*. Number of distinct variants in study B.
- study.studyId: *Character vector*. Study ID from study list.
- study.traitReported: *Character vector*. Reported trait from study list.
- study.traitCategory: *Character vector*. Trait category from study list.

The variant intersection set (variant\_intersection\_set) is a character vector representing the intersection of variants.

**Examples**

```
## Not run:
result <- overlapInfoForStudy(study_id = "GCST90002357",
  study_ids = list("GCST90025975", "GCST90025962"))

## End(Not run)
```

---

pathwaysQuery

*Retrieve Pathways data for a specified gene.*


---

**Description**

This function queries the Open Targets GraphQL API to retrieve pathways data for a specified gene.

**Usage**

```
pathwaysQuery(ensgId)
```

**Arguments**

ensgId                      Character: ENSEMBL ID of the target gene (e.g., ENSG00000105397).

**Value**

Returns a tibble containing pathways data for the specified gene.

**Examples**

```
## Not run:  
result <- pathwaysQuery(ensgId = "ENSG00000105397")  
  
## End(Not run)
```

---

pharmacogenomicsChemblQuery

*Retrieve Pharmacogenomics data for a specified drug.*

---

**Description**

This function queries the Open Targets GraphQL API to retrieve pharmacogenomics data for a specified drug.

**Usage**

```
pharmacogenomicsChemblQuery(chemblId)
```

**Arguments**

chemblId                      Character: ChEMBL ID of the target drug (e.g., "CHEMBL1016").

**Value**

Returns a tibble containing pharmacogenomics data for the specified drug.

**Examples**

```
## Not run:  
result <- pharmacogenomicsChemblQuery(chemblId = "CHEMBL1016")  
  
## End(Not run)
```

---

`pharmacogenomicsGeneQuery`*Retrieve Pharmacogenomics data for a specified gene.*

---

**Description**

This function queries the Open Targets GraphQL API to retrieve pharmacogenomics data for a specified gene.

**Usage**`pharmacogenomicsGeneQuery(ensgId)`**Arguments**

`ensgId` Character: ENSEMBL ID of the target gene (e.g., ENSG00000141510).

**Value**

Returns a tibble containing pharmacogenomics data for the specified gene.

**Examples**

```
## Not run:  
result <- pharmacogenomicsGeneQuery(ensgId = "ENSG00000141510")  
  
## End(Not run)
```

---

`pharmacogenomicsVariantQuery`*Retrieve Pharmacogenomics data for a specified variant.*

---

**Description**

This function queries the Open Targets GraphQL API to retrieve pharmacogenomics data for a specified variant.

**Usage**`pharmacogenomicsVariantQuery(variantId)`**Arguments**

`variantId` Character: ID of the target variant (e.g., "12\_111446804\_T\_C").

**Value**

Returns a tibble containing pharmacogenomics data for the specified variant.

**Examples**

```
## Not run:  
result <- pharmacogenomicsVariantQuery(variantId = "12_111446804_T_C")  
  
## End(Not run)
```

---

qtlCredibleSetsQuery    *Retrieve QTL Credible Sets data for a specified variant.*

---

**Description**

This function queries the Open Targets GraphQL API to retrieve QTL credible sets data for a specified variant.

**Usage**

```
qtlCredibleSetsQuery(variantId, size = 500, index = 0)
```

**Arguments**

|           |                                                                |
|-----------|----------------------------------------------------------------|
| variantId | Character: ID of the target variant (e.g., "19_10352442_G_C"). |
| size      | Integer: Number of records to retrieve (default: 500).         |
| index     | Integer: Page index for pagination (default: 0).               |

**Value**

Returns a tibble containing QTL credible sets data for the specified variant.

**Examples**

```
## Not run:  
result <- qtlCredibleSetsQuery(variantId = "19_10352442_G_C", size = 500,  
index = 0)  
  
## End(Not run)
```

---

`safetyQuery`*Retrieve Safety Liabilities data for a specified gene.*

---

**Description**

This function queries the Open Targets GraphQL API to retrieve safety liabilities data for a specified gene.

**Usage**

```
safetyQuery(ensgId)
```

**Arguments**

`ensgId` Character: ENSEMBL ID of the target gene (e.g., ENSG00000141510).

**Value**

Returns a tibble containing safety liabilities data for the specified gene.

**Examples**

```
## Not run:
result <- safetyQuery(ensgId = "ENSG00000141510")

## End(Not run)
```

---

`sharedTraitStudiesQuery`*Retrieve Shared Trait Studies data for specified diseases.*

---

**Description**

This function queries the Open Targets GraphQL API to retrieve shared trait studies data for specified disease IDs.

**Usage**

```
sharedTraitStudiesQuery(diseaseIds, size = 500, index = 0)
```

**Arguments**

`diseaseIds` Character vector: IDs of the target diseases (e.g., `c("EFO_0004587")`).  
`size` Integer: Number of records to retrieve (default: 500).  
`index` Integer: Page index for pagination (default: 0).

**Value**

Returns a tibble containing shared trait studies data for the specified diseases.

**Examples**

```
## Not run:
result <- sharedTraitStudiesQuery(diseaseIds = c("EFO_0004587"), size = 500,
index = 0)

## End(Not run)
```

---

uniprotLiteratureQuery

*Retrieve UniProt Literature data for a specified gene and disease.*

---

**Description**

This function queries the Open Targets GraphQL API to retrieve UniProt literature evidence data for a specified gene and disease.

**Usage**

```
uniprotLiteratureQuery(ensemblId, efoId, size = 3500)
```

**Arguments**

|           |                                                                     |
|-----------|---------------------------------------------------------------------|
| ensemblId | Character: ENSEMBL ID of the target gene (e.g., "ENSG00000130203"). |
| efoId     | Character: EFO ID of the target disease (e.g., "MONDO_0004975").    |
| size      | Integer: Number of records to retrieve (default: 3500).             |

**Value**

Returns a tibble containing UniProt literature evidence data for the specified gene and disease.

**Examples**

```
## Not run:
result <- uniprotLiteratureQuery(ensemblId = "ENSG00000130203", efoId =
"MONDO_0004975", size = 3500)

## End(Not run)
```



---

uniProtVariantsQuery    *Retrieve UniProt Variants data for a specified variant.*

---

**Description**

This function queries the Open Targets GraphQL API to retrieve UniProt variants data for a specified variant.

**Usage**

```
uniProtVariantsQuery(variantId)
```

**Arguments**

variantId            Character: ID of the target variant (e.g., "4\_1804392\_G\_A").

**Value**

Returns a tibble containing UniProt variants data for the specified variant.

**Examples**

```
## Not run:
result <- uniProtVariantsQuery(variantId = "4_1804392_G_A")

## End(Not run)
```

---

variantEffectPredictorQuery  
                          *Retrieve Variant Effect Predictor data for a specified variant.*

---

**Description**

This function queries the Open Targets GraphQL API to retrieve variant effect predictor data for a specified variant.

**Usage**

```
variantEffectPredictorQuery(variantId)
```

**Arguments**

variantId            Character: ID of the target variant (e.g., "4\_1804392\_G\_A").

**Value**

Returns a tibble containing variant effect predictor data for the specified variant.

**Examples**

```
## Not run:
result <- variantEffectPredictorQuery(variantId = "4_1804392_G_A")

## End(Not run)
```

---

|                    |                                                              |
|--------------------|--------------------------------------------------------------|
| variantEffectQuery | <i>Retrieve Variant Effect data for a specified variant.</i> |
|--------------------|--------------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve variant effect data for a specified variant.

**Usage**

```
variantEffectQuery(variantId)
```

**Arguments**

variantId      Character: ID of the target variant (e.g., "4\_1804392\_G\_A").

**Value**

Returns a tibble containing variant effect data for the specified variant.

**Examples**

```
## Not run:
result <- variantEffectQuery(variantId = "4_1804392_G_A")

## End(Not run)
```

---

|               |                                                            |
|---------------|------------------------------------------------------------|
| variantsQuery | <i>Retrieve Variants data for a specified study locus.</i> |
|---------------|------------------------------------------------------------|

---

**Description**

This function queries the Open Targets GraphQL API to retrieve variants data for a specified study locus.

**Usage**

```
variantsQuery(studyLocusId, size = 500, index = 0)
```

**Arguments**

|              |                                                                                     |
|--------------|-------------------------------------------------------------------------------------|
| studyLocusId | Character: ID of the target study locus (e.g., "fa375739ca2a6b825ce5cc69d117e84b"). |
| size         | Integer: Number of records to retrieve (default: 500).                              |
| index        | Integer: Page index for pagination (default: 0).                                    |

**Value**

Returns a tibble containing variants data for the specified study locus.

**Examples**

```
## Not run:  
result <- variantsQuery(studyLocusId = "fa375739ca2a6b825ce5cc69d117e84b",  
  size = 500, index = 0)  
  
## End(Not run)
```

# Index

adverseEventsQuery, [2](#)

chemblQuery, [3](#)  
clinVarQuery, [4](#)  
compGenomicsQuery, [5](#)

depMapQuery, [5](#)

europePMCQuery, [6](#)

geneBurdenQuery, [7](#)  
geneOntologyQuery, [7](#)  
geneticConstraintQuery, [8](#)  
genomicsEnglandQuery, [9](#)  
gwasColocalisation, [9](#)  
gwasCredibleSet, [11](#)  
gwasCredibleSetsQuery, [12](#)

hallmarksQuery, [12](#)

indicationsQuery, [13](#)  
interactionsQuery, [14](#)

knownDrugsChemblQuery, [14](#)  
knownDrugsGeneQuery, [15](#)

locus2GeneQuery, [16](#)

mechanismsOfActionQuery, [16](#)  
mousePhenotypesQuery, [17](#)

orphanetQuery, [18](#)  
overlapInfoForStudy, [18](#)

pathwaysQuery, [19](#)  
pharmacogenomicsChemblQuery, [20](#)  
pharmacogenomicsGeneQuery, [21](#)  
pharmacogenomicsVariantQuery, [21](#)

qtlCredibleSetsQuery, [22](#)

safetyQuery, [23](#)

sharedTraitStudiesQuery, [23](#)

uniprotLiteratureQuery, [24](#)  
uniProtVariantsQuery, [25](#)

variantEffectPredictorQuery, [25](#)  
variantEffectQuery, [26](#)  
variantsQuery, [26](#)