Package 'CohortDiagnostics'

September 10, 2021

Type Package

Title Diagnostics for OHDSI Cohorts

Version 2.2.0

Date 2021-09-10

Maintainer Gowtham Rao <rao@ohdsi.org>

Description Diagnostics for cohorts that use the OMOP Common Data Model and the OHDSI tools.

Depends DatabaseConnector (>= 4.0.0),
FeatureExtraction (>= 3.1.1),
R (>= 4.0.0)

Imports Andromeda (>= 0.5.0),
checkmate,
clock,
digest,
dplyr (>= 1.0.0),
methods,
ParallelLogger (>= 2.0.2),
readr (>= 2.0.1),
RJSONIO,
rlang,
ROhdsiWebApi (>= 1.2.0),
SqlRender (>= 1.7.0),
stringr,
tidyr (>= 1.1.3),
tsibble

Suggests CirceR,
DT,
Eunomia,
ggiraph,
ggplot2,
htmltools,
knitr,
lubridate,
plotly,
pool,
purrr,
RColorBrewer,
markdown,
rmarkdown,

RSQLite (>= 2.2.1),
scales,
shiny,
shinydashboard,
shinyWidgets,
testthat,
withr,
zip,
R.utils

Remotes ohdsi/Eunomia,
ohdsi/FeatureExtraction,
ohdsi/ROhdsiWebApi,
ohdsi/CirceR

License Apache License

VignetteBuilder knitr

URL https://ohdsi.github.io/CohortDiagnostics, https://github.com/OHDSI/CohortDiagnostics

BugReports https://github.com/OHDSI/CohortDiagnostics/issues

RoxygenNote 7.1.1

Encoding UTF-8

Language en-US

R topics documented:

`checkIfCohortInstantiated` *Checks if a set of cohortId(s) are instantiated in the cohort table*

Description

Given a set of one or more cohortIds and a single cohort table, checks if all cohortIds in the set are instantiated.

Usage

```
checkIfCohortInstantiated(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema,
  cohortTable,
  cohortIds
)
```

Arguments

`connectionDetails`	An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package. Can be left NULL if `connection` is provided.
`connection`	An object of type `connection` as created using the `connect` function in the DatabaseConnector package. Can be left NULL if `connectionDetails` is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
`cohortDatabaseSchema`	Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
`cohortTable`	Name of the cohort table.
`cohortIds`	Provide a set of cohort IDs to check if instantiated.

Value

Returns TRUE if all cohortIds are instantiated.

`checkInputFileEncoding` *Check character encoding of input file*

Description

For its input files, CohortDiagnostics only accepts UTF-8 or ASCII character encoding. This function can be used to check whether a file meets these criteria.

Usage

```
checkInputFileEncoding(fileName)
```

Arguments

`fileName`	The path to the file to check

Value

Throws an error if the input file does not have the correct encoding.

createDatabaseDataSource *Return a database data source object*

Description

Collects a list of objects needed to connect to a database datsource. This includes one of DatabaseConnector::createCo object, or a DBI database connection created using either DatabaseConnector::connection or pool::dbPool, and a names of resultsDatabaseSchema and vocabularyDatabaseSchema

Usage

```
createDatabaseDataSource(
  connection = NULL,
  connectionDetails = NULL,
  resultsDatabaseSchema,
  vocabularyDatabaseSchema = resultsDatabaseSchema
)
```

Arguments

connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
resultsDatabaseSchema	Schema name where the output of your Cohort Diagnostics result set it uploaded. This is commonly uploaded using CohortDiagnostics::uploadResults function. The schema may be initiated using CohortDiagnostics::resultsDatabaseSchema.
vocabularyDatabaseSchema	Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.

Value

Returns a list with information on database data source

createFileDataSource *Return a file data source object*

Description

Given a premerged file (an .RData/rds object the output of `CohortDiagnostics::preMergeDiagnosticsFiles` reads the object into memory and makes it available for query.

Usage

```
createFileDataSource(premergedDataFile, envir = .GlobalEnv)
```

Arguments

`premergedDataFile`	an .RData/rds object the output of `CohortDiagnostics::preMergeDiagnosticsFiles`
`envir`	(optional) R-environment to read premerged data. By default this is the global environment.

Value

R environment containing data conforming to Cohort Diagnostics results data model specifications.

createResultsDataModel *Create the results data model tables on a database server.*

Description

Create the results data model tables on a database server.

Usage

```
createResultsDataModel(connection = NULL, connectionDetails = NULL, schema)
```

Arguments

`connection`	An object of type `connection` as created using the `connect` function in the DatabaseConnector package. Can be left NULL if `connectionDetails` is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
`connectionDetails`	An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package. Can be left NULL if `connection` is provided.
`schema`	The schema on the postgres server where the tables will be created.

Details

Only PostgreSQL servers are supported.

exportFeatureExtractionOutput
Export Feature Extraction output to csv

Description

Exports the output of `FeatureExtraction::getDbCovariateData` into CSV.

Usage

```
exportFeatureExtractionOutput(
  featureExtractionDbCovariateData,
  databaseId,
  incremental = FALSE,
  covariateValueFileName = "covariate_value.csv",
  covariateValueContFileName = "covariate_value_dist.csv",
  covariateRefFileName = "covariate_ref.csv",
  analysisRefFileName = "analysis_ref.csv",
  timeDistributionFileName = NULL,
  timeRefFileName = NULL,
  cohortCounts,
  minCellCount = 5
)
```

Arguments

`featureExtractionDbCovariateData`	An Andromeda object returned by `CohortDiagonstics::runCohortCharacterizationDiagnosti`
`databaseId`	A short string for identifying the database (e.g. 'Synpuf').
`incremental`	Create only cohort diagnostics that haven't been created before?
`covariateValueFileName`	The full path (including file name) for the csv file with covariate value data. e.g. "covariate_value.csv" or "temporal_covariate_value.csv"
`covariateValueContFileName`	The full path (including file name) for the csv file with covariate value distribution data. e.g. "covariate_value_dist.csv" or "temporal_covariate_value_dist.csv"
`covariateRefFileName`	The full path (including file name) for the csv file with covariate reference data. e.g. "covariate_ref.csv" or "temporal_covariate_ref.csv"
`analysisRefFileName`	The full path (including file name) for the csv file with analysis reference data. e.g. "analysis_ref.csv" or "temporal_analysis_ref.csv"
`timeDistributionFileName`	The full path (including file name) for the csv file with time distribution data. e.g. "time_distribution.csv"
`timeRefFileName`	The full path (including file name) for the csv file with time reference data. e.g. "temporal_time_ref.csv"
`cohortCounts`	Output `CohortDiagnostics::getCohortCounts`

`minCellCount` (Optional). Default value = 5. The minimum cell count for fields contains person counts or fractions.

`getCdmDataSourceInformation`

Returns information from CDM source table.

Description

Returns CDM source name, description, release date, CDM release date, version and vocabulary version, where available.

Usage

```
getCdmDataSourceInformation(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema
)
```

Arguments

`connectionDetails`
: An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package. Can be left NULL if `connection` is provided.

`connection`
: An object of type `connection` as created using the `connect` function in the DatabaseConnector package. Can be left NULL if `connectionDetails` is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.

`cdmDatabaseSchema`
: Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.

Value

Returns a data frame from CDM Data source.

`getCirceRenderedExpression`

Returns list with circe generated documentation

Description

Returns list with circe generated documentation

Usage

```
getCirceRenderedExpression(cohortDefinition)
```

Arguments

`cohortDefinition`
: An object with a list representation of the cohort definition expression.

Value

list object

`getCohortAsFeatureTemporalCharacterizationResults`
Returns cohort temporal feature characterization

Description

Returns a list object with temporalCovariateValue, temporalCovariateRef, temporalAnalysisRef, temporalRef output of cohort as features.

Usage

```
getCohortAsFeatureTemporalCharacterizationResults(
  dataSource = .GlobalEnv,
  cohortIds = NULL,
  databaseIds = NULL,
  temporalTimeRef = getResultsTemporalTimeRef(dataSource = dataSource)
)
```

Arguments

`dataSource`
: A list object that is the output of `createDatabaseDataSource` or `createFileDataSource` function. This object helps direct the function to query data from the database (created by `createDatabaseDataSource`) or a local premerged file (created by `createFileDataSource`). Premerged files are output of cohortDiagnostics compiled into RData using `preMergeDiagnosticsFiles`. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function `createResultsDataModel` and uploaded using `uploadResults`

`cohortIds`
: A vector of one or more Cohort Ids.

`databaseIds`
: A vector one or more databaseIds to retrieve the results for. This is a character field values from the ‘databaseId‘ field of the ‘database‘ table of the results data model.

`temporalTimeRef`
: A dataframe object with three columns timeId (integer), startDay (integer), endDay (integer)

Value

Returns a list object with temporalCovariateValue, temporalCovariateRef, temporalAnalysisRef, temporalRef output of cohort as features.

getCohortCounts *Count of unique subjects and records in the cohort(s)*

Description

Computes the subject and entry count per cohort

Usage

```
getCohortCounts(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema,
  cohortTable = "cohort",
  cohortIds = c()
)
```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
cohortDatabaseSchema	Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
cohortTable	Name of the cohort table.
cohortIds	The cohort Id(s) used to reference the cohort in the cohort table. If left empty, all cohorts in the table will be included.

Value

A tibble with cohort counts

getCohortOverlap *Returns data for use in cohort_overlap*

Description

Returns data for use in cohort_overlap

Usage

```
getCohortOverlap(dataSource, cohortIds = NULL, databaseIds = NULL)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the ‘databaseId‘ field of the ‘database‘ table of the results data model.

Value

Returns data for use in cohort_overlap

---

getCohortRelationshipCharacterizationResults

Returns cohort as feature characterization

Description

Returns a list object with covariateValue, covariateRef, analysisRef output of cohort as features.

Usage

```
getCohortRelationshipCharacterizationResults(
  dataSource = .GlobalEnv,
  cohortIds = NULL,
  databaseIds = NULL
)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the ‘databaseId‘ field of the ‘database‘ table of the results data model.

Value

Returns a list object with covariateValue, covariateRef, analysisRef output of cohort as features. To avoid clash with covaraiteId and conceptId returned from Feature Extraction the output is a negative integer.

`getConcept` *Returns conceptIds details from concept table*

Description

Returns concept details from concept table for provided list of concept ids

Usage

```
getConcept(
  dataSource = .GlobalEnv,
  vocabularyDatabaseSchema = NULL,
  conceptIds = NULL
)
```

Arguments

`dataSource`	A list object that is the output of `createDatabaseDataSource` or `createFileDataSource` function. This object helps direct the function to query data from the database (created by `createDatabaseDataSource`) or a local premerged file (created by `createFileDataSource`). Premerged files are output of cohortDiagnostics compiled into RData using `preMergeDiagnosticsFiles`. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function `createResultsDataModel` and uploaded using `uploadResults`
`vocabularyDatabaseSchema`	Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.
`conceptIds`	A vector of one or more Concept Ids.

Value

Returns a data frame (tibble)

getConceptAncestor *Returns data from concept ancestor table for vector of concept ids*

Description

Returns data from concept ancestor table for vector of concept ids

Usage

```
getConceptAncestor(
  dataSource = .GlobalEnv,
  vocabularyDatabaseSchema = NULL,
  conceptIds = NULL
)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
vocabularyDatabaseSchema	Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.
conceptIds	a vector of concept ids

Value

Returns a data frame (tibble)

getConceptMetadata *Returns a metadata for concept ids*

Description

Returns a metadata for a given list of concept ids that includes concept synonyms, concept relationship, concept ancestor, concept count per database, concept cooccurrence on index date per database and cohortId.

Usage

```
getConceptMetadata(
  dataSource,
  databaseIds = NULL,
  cohortIds = NULL,
  vocabularyDatabaseSchema = NULL,
  conceptIds = NULL,
  getConceptRelationship = TRUE,
  getConceptAncestor = TRUE,
  getConceptSynonym = TRUE,
  getConceptCount = TRUE,
  getConceptCooccurrence = TRUE,
  getIndexEventCount = TRUE,
  getConceptMappingCount = TRUE
)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.
cohortIds	A vector of one or more Cohort Ids.
vocabularyDatabaseSchema	Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.
conceptIds	A vector of one or more Concept Ids.
getConceptRelationship	Do you want conceptRelationship?
getConceptAncestor	Do you want conceptAncestor?
getConceptSynonym	Do you want conceptSynonym?
getConceptCount	Do you want conceptCount?
getConceptCooccurrence	Do you want concept cooccurrence?
getIndexEventCount	Do you want index event concept count?
getConceptMappingCount	Do you want concept mapping count?

Value

Returns a list of data frames (tibbles)

`getConceptRelationship` *Returns data from concept relationship table for list of concept ids*

Description

Returns data from concept relationship table for list of concept ids

Usage

```
getConceptRelationship(
  dataSource = .GlobalEnv,
  vocabularyDatabaseSchema = NULL,
  conceptIds = NULL
)
```

Arguments

`dataSource`	A list object that is the output of `createDatabaseDataSource` or `createFileDataSource` function. This object helps direct the function to query data from the database (created by `createDatabaseDataSource`) or a local premerged file (created by `createFileDataSource`). Premerged files are output of cohortDiagnostics compiled into RData using `preMergeDiagnosticsFiles`. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function `createResultsDataModel` and uploaded using `uploadResults`
`vocabularyDatabaseSchema`	Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.
`conceptIds`	A vector of one or more Concept Ids.

Value

Returns a data frame (tibble)

getConceptSynonym *Returns data from concept synonym table for list of concept ids*

Description

Returns data from concept synonym table for list of concept ids

Usage

```
getConceptSynonym(
  dataSource = .GlobalEnv,
  vocabularyDatabaseSchema = NULL,
  conceptIds = NULL
)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
vocabularyDatabaseSchema	Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.
conceptIds	A vector of one or more Concept Ids.

Value

Returns a data frame (tibble)

getDomainInformation *Get domain information*

Description

Get domain information

Usage

```
getDomainInformation(packageName = NULL)
```

Arguments

packageName	e.g. 'CohortDiagnostics'

Value

A list with two tibble data frame objects with domain information represented in wide and long format respectively.

`getFeatureExtractionCharacterization` *Returns cohort characterization output of feature extraction*

Description

Returns a list object with covariateValue, covariateValueDist, covariateRef, analysisRef output of feature extraction.

Usage

```
getFeatureExtractionCharacterization(
  dataSource = .GlobalEnv,
  cohortIds = NULL,
  databaseIds = NULL
)
```

Arguments

`dataSource`	A list object that is the output of `createDatabaseDataSource` or `createFileDataSource` function. This object helps direct the function to query data from the database (created by `createDatabaseDataSource`) or a local premerged file (created by `createFileDataSource`). Premerged files are output of cohortDiagnostics compiled into RData using `preMergeDiagnosticsFiles`. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function `createResultsDataModel` and uploaded using `uploadResults`
`cohortIds`	A vector of one or more Cohort Ids.
`databaseIds`	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a list object with covariateValue, covariateValueDist, covariateRef, analysisRef output of feature extraction along with concept information.

getFeatureExtractionTemporalCharacterization

Returns temporal cohort characterization output of feature extraction

Description

Returns a list object with temporalCovariateValue, temporalCovariateValueDist, temporalCovariateRef, temporalAnalysisRef, temporalRef output of feature extraction along with concept information.

Usage

```
getFeatureExtractionTemporalCharacterization(
  dataSource = .GlobalEnv,
  cohortIds = NULL,
  databaseIds = NULL
)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a list object with temporalCovariateValue, temporalCovariateValueDist, temporalCovariateRef, temporalAnalysisRef, temporalTimeRef, Concept output of feature extraction.

getMultipleCharacterizationResults

Returns multiple characterization output

Description

Returns multiple characterization output

Usage

```
getMultipleCharacterizationResults(
  dataSource = .GlobalEnv,
  cohortIds = NULL,
  databaseIds = NULL
)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns multiple characterization output

---

getOptimizationRecommendationForConceptSetExpression

given a concept set table, get optimization recommendation

Description

given a concept set table, get optimization recommendation

Usage

```
getOptimizationRecommendationForConceptSetExpression(
  conceptSetExpression,
  vocabularyDatabaseSchema = "vocabulary",
  tempEmulationSchema = tempEmulationSchema,
  connectionDetails = NULL,
  connection = NULL
)
```

Arguments

conceptSetExpression	An R Object (list) with concept set expression. This maybe generated by first getting the JSON representation of concept set expression and converting it to a list using RJSONIO::fromJson(digits = 23)

vocabularyDatabaseSchema
: Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.

tempEmulationSchema
: Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

connectionDetails
: An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.

connection
: An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.

---

getResultsCohort *Returns data from cohort table of Cohort Diagnostics results data model*

Description

Returns data from cohort table of Cohort Diagnostics results data model

Usage

```
getResultsCohort(dataSource)
```

Arguments

dataSource
: A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults

Value

Returns a data frame (tibble)

getResultsCohortCount *Returns data from cohort_count table of Cohort Diagnostics results data model*

Description

Returns data from cohort_count table of Cohort Diagnostics results data model

Usage

```
getResultsCohortCount(dataSource, cohortIds = NULL, databaseIds = NULL)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a data frame (tibble)

getResultsCohortInclusion *Returns data from cohort_inclusion table of Cohort Diagnostics results data model*

Description

Returns data from cohort_inclusion table of Cohort Diagnostics results data model

Usage

```
getResultsCohortInclusion(dataSource, cohortIds = NULL, databaseIds = NULL)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a data frame (tibble).

---

getResultsCohortInclusionStats

Returns data from cohort_inclusion_stats table of Cohort Diagnostics results data model

---

Description

Returns data from cohort_inclusion_stats table of Cohort Diagnostics results data model

Usage

```
getResultsCohortInclusionStats(
  dataSource,
  cohortIds = NULL,
  databaseIds = NULL
)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a data frame (tibble).

`getResultsCohortRelationships`

Returns data from cohort_relationships table of Cohort Diagnostics results data model

---

Description

Returns data from cohort_relationships table of Cohort Diagnostics results data model

Usage

```
getResultsCohortRelationships(dataSource, cohortIds = NULL, databaseIds = NULL)
```

Arguments

`dataSource`	A list object that is the output of `createDatabaseDataSource` or `createFileDataSource` function. This object helps direct the function to query data from the database (created by `createDatabaseDataSource`) or a local premerged file (created by `createFileDataSource`). Premerged files are output of cohortDiagnostics compiled into RData using `preMergeDiagnosticsFiles`. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function `createResultsDataModel` and uploaded using `uploadResults`
`cohortIds`	A vector of one or more Cohort Ids.
`databaseIds`	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a data frame (tibble) with results that conform to cohort_relationships table in Cohort Diagnostics results data model.

`getResultsCohortSummaryStats`

Returns data from cohort_summary_stats table of Cohort Diagnostics results data model

---

Description

Returns data from cohort_summary_stats table of Cohort Diagnostics results data model

Usage

```
getResultsCohortSummaryStats(dataSource, cohortIds = NULL, databaseIds = NULL)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a data frame (tibble).

---

getResultsConceptCooccurrence

Returns concept cooccurrence for a list of cohortIds and databaseIds combinations

---

Description

Given a list of cohortIds, databaseIds combinations the function returns precomputed concept cooccurrence conceptIds for the combination.

Usage

```
getResultsConceptCooccurrence(dataSource, databaseIds = NULL, cohortIds = NULL)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.
cohortIds	A vector of one or more Cohort Ids.

Value

Returns a data frame (tibble)

getResultsConceptCount *Returns data from concept_count table of Cohort Diagnostics results data model*

Description

Returns data from concept_count table of Cohort Diagnostics results data model

Usage

```
getResultsConceptCount(dataSource, databaseIds = NULL, conceptIds = NULL)
```

Arguments

dataSource	A list object that is the output of `createDatabaseDataSource` or `createFileDataSource` function. This object helps direct the function to query data from the database (created by `createDatabaseDataSource`) or a local premerged file (created by `createFileDataSource`). Premerged files are output of cohortDiagnostics compiled into RData using `preMergeDiagnosticsFiles`. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function `createResultsDataModel` and uploaded using `uploadResults`
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.
conceptIds	A list of concept ids to get counts for

Value

Returns a data frame (tibble)

getResultsConceptCountSummary *Returns summary data from concept_count table of Cohort Diagnostics results data model*

Description

Returns summary data from concept_count table of Cohort Diagnostics results data model

Usage

```
getResultsConceptCountSummary(
  dataSource,
  databaseIds,
  conceptIds,
  minDate = NULL,
  maxDate = NULL
)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.
conceptIds	A list of concept ids to get counts for
minDate	Minimum date of range
maxDate	Maximum date of range

Value

Returns a data frame (tibble)

---

getResultsConceptSubjects

Returns data from concept_subjects table of Cohort Diagnostics results data model

---

Description

Returns data from concept_subjects table of Cohort Diagnostics results data model

Usage

```
getResultsConceptSubjects(dataSource, databaseIds = NULL, conceptIds = NULL)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.
conceptIds	A list of concept ids to get counts for

Value

Returns a data frame (tibble)

getResultsDataModelSpecifications

Get specifications for Cohort Diagnostics results data model

Description

Get specifications for Cohort Diagnostics results data model

Usage

```
getResultsDataModelSpecifications(versionNumber = NULL, packageName = NULL)
```

Arguments

versionNumber	Which version of Cohort Diagnostics. Default will be the most recent version.
packageName	e.g. 'CohortDiagnostics'

Value

A tibble data frame object with specifications

getResultsExcludedConcepts

Returns excluded concepts for a list of cohortIds and databaseIds combinations

Description

Given a list of cohortIds, databaseIds combinations the function returns precomputed excluded conceptIds for the combination.

Usage

```
getResultsExcludedConcepts(dataSource, databaseIds = NULL, cohortIds = NULL)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.
cohortIds	A vector of one or more Cohort Ids.

Value

Returns a data frame (tibble)

`getResultsFixedTimeSeries` *Returns data from time_series table of Cohort Diagnostics results data model*

Description

Returns data from time_series table of Cohort Diagnostics results data model. The returned object is a tsibble, but to use in time series analysis, gaps need to be filled. Only absolute values are returned i.e. negative values are converted to positives.

Usage

```
getResultsFixedTimeSeries(dataSource, cohortIds = NULL, databaseIds = NULL)
```

Arguments

`dataSource`	A list object that is the output of `createDatabaseDataSource` or `createFileDataSource` function. This object helps direct the function to query data from the database (created by `createDatabaseDataSource`) or a local premerged file (created by `createFileDataSource`). Premerged files are output of cohortDiagnostics compiled into RData using `preMergeDiagnosticsFiles`. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function `createResultsDataModel` and uploaded using `uploadResults`
`cohortIds`	A vector of one or more Cohort Ids.
`databaseIds`	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a list of tsibble (time series) objects with results that conform to time series table in Cohort Diagnostics results data model. There are three list objects, labeled m for monthly, q for quarterly and y for yearly. The periodBegin variable is in the format of tsibble::yearmonth for monthly, tsibble::yearquarter for quarter and integer for year.

getResultsIncidenceRate

Returns data from incidence_rate table of Cohort Diagnostics results data model

Description

Returns data from incidence_rate table of Cohort Diagnostics results data model

Usage

```
getResultsIncidenceRate(dataSource, cohortIds = NULL, databaseIds = NULL)
```

Arguments

dataSource	A list object that is the output of `createDatabaseDataSource` or `createFileDataSource` function. This object helps direct the function to query data from the database (created by `createDatabaseDataSource`) or a local premerged file (created by `createFileDataSource`). Premerged files are output of cohortDiagnostics compiled into RData using `preMergeDiagnosticsFiles`. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function `createResultsDataModel` and uploaded using `uploadResults`
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a data frame (tibble).

getResultsInclusionRuleStatistics

Returns data from inclusion_rule_stats table of Cohort Diagnostics results data model

Description

Returns data from inclusion_rule_stats table of Cohort Diagnostics results data model

Usage

```
getResultsInclusionRuleStatistics(
  dataSource,
  cohortIds = NULL,
  databaseIds = NULL
)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a data frame (tibble).

---

getResultsIndexEventBreakdown

Returns data from index_event_breakdown table of Cohort Diagnostics results data model

---

Description

Returns data from index_event_breakdown table of Cohort Diagnostics results data model

Usage

```
getResultsIndexEventBreakdown(dataSource, cohortIds = NULL, databaseIds = NULL)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a data frame (tibble) with results that conform to index_event_breakdown table in Cohort Diagnostics results data model.

getResultsMetadata *Returns data from meta data table of Cohort Diagnostics results data model*

Description

Returns data from meta data table of Cohort Diagnostics results data model

Usage

```
getResultsMetadata(dataSource)
```

Arguments

dataSource	A list object that is the output of `createDatabaseDataSource` or `createFileDataSource` function. This object helps direct the function to query data from the database (created by `createDatabaseDataSource`) or a local premerged file (created by `createFileDataSource`). Premerged files are output of cohortDiagnostics compiled into RData using `preMergeDiagnosticsFiles`. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function `createResultsDataModel` and uploaded using `uploadResults`

Value

Returns a data frame (tibble)

getResultsOrphanConcept *Returns data from orphan_concept table of Cohort Diagnostics results data model*

Description

Returns data from orphan_concept table of Cohort Diagnostics results data model

Usage

```
getResultsOrphanConcept(dataSource, cohortIds = NULL, databaseIds = NULL)
```

Arguments

dataSource	A list object that is the output of `createDatabaseDataSource` or `createFileDataSource` function. This object helps direct the function to query data from the database (created by `createDatabaseDataSource`) or a local premerged file (created by `createFileDataSource`). Premerged files are output of cohortDiagnostics compiled into RData using `preMergeDiagnosticsFiles`. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function `createResultsDataModel` and uploaded using `uploadResults`

cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a data frame (tibble).

`getResultsResolvedConcepts`

Returns resolved concepts for a list of cohortIds and databaseIds combinations

---

Description

Given a list of cohortIds, databaseIds combinations the function returns precomputed resolved conceptIds for the combination.

Usage

```
getResultsResolvedConcepts(dataSource, databaseIds = NULL, cohortIds = NULL)
```

Arguments

dataSource	A list object that is the output of `createDatabaseDataSource` or `createFileDataSource` function. This object helps direct the function to query data from the database (created by `createDatabaseDataSource`) or a local premerged file (created by `createFileDataSource`). Premerged files are output of cohortDiagnostics compiled into RData using `preMergeDiagnosticsFiles`. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function `createResultsDataModel` and uploaded using `uploadResults`
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.
cohortIds	A vector of one or more Cohort Ids.

Value

Returns a data frame (tibble)

getResultsTimeDistribution

Returns data from time_distribution table of Cohort Diagnostics results data model

Description

Returns data from time_distribution table of Cohort Diagnostics results data model

Usage

```
getResultsTimeDistribution(dataSource, cohortIds = NULL, databaseIds = NULL)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a data frame (tibble).

getResultsVisitContext

Returns data from visit_context table of Cohort Diagnostics results data model

Description

Returns data from visit_context table of Cohort Diagnostics results data model

Usage

```
getResultsVisitContext(dataSource, cohortIds = NULL, databaseIds = NULL)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults
cohortIds	A vector of one or more Cohort Ids.
databaseIds	A vector one or more databaseIds to retrieve the results for. This is a character field values from the 'databaseId' field of the 'database' table of the results data model.

Value

Returns a data frame (tibble) with results that conform to visit_context table in Cohort Diagnostics results data model.

---

getVocabularyRelationship

Returns data from relationship table of Cohort Diagnostics results data model

---

Description

Returns data from relationship table of Cohort Diagnostics results data model

Usage

```
getVocabularyRelationship(dataSource)
```

Arguments

dataSource	A list object that is the output of createDatabaseDataSource or createFileDataSource function. This object helps direct the function to query data from the database (created by createDatabaseDataSource) or a local premerged file (created by createFileDataSource). Premerged files are output of cohortDiagnostics compiled into RData using preMergeDiagnosticsFiles. Database DataSources are data inserted into a remote database (only a postgres database is supported) with tables created with DDL function createResultsDataModel and uploaded using uploadResults

Value

Returns a data frame (tibble)

`instantiateCohortSet` *Instantiate a set of cohort(s)*

Description

This function instantiates a set of cohort(s) in specified cohort table, using definitions that are fetched from a WebApi interface. Optionally, the inclusion rule statistics are computed and stored in the `inclusionStatisticsFolder`.

Usage

```
instantiateCohortSet(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  vocabularyDatabaseSchema = cdmDatabaseSchema,
  tempEmulationSchema = NULL,
  oracleTempSchema = NULL,
  cohortDatabaseSchema = cdmDatabaseSchema,
  cohortTable = "cohort",
  cohortIds = NULL,
  packageName = NULL,
  cohortToCreateFile = "settings/CohortsToCreate.csv",
  baseUrl = NULL,
  cohortSetReference = NULL,
  generateInclusionStats = FALSE,
  inclusionStatisticsFolder = NULL,
  createCohortTable = TRUE,
  incremental = FALSE,
  incrementalFolder = NULL
)
```

Arguments

`connectionDetails`
: An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package. Can be left NULL if `connection` is provided.

`connection`
: An object of type `connection` as created using the `connect` function in the DatabaseConnector package. Can be left NULL if `connectionDetails` is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.

`cdmDatabaseSchema`
: Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.

`vocabularyDatabaseSchema`
: Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.

`tempEmulationSchema`
: Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

`oracleTempSchema`
: DEPRECATED by DatabaseConnector: use `tempEmulationSchema` instead.

`cohortDatabaseSchema`
: Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

`cohortTable`
: Name of the cohort table.

`cohortIds`
: Optionally, provide a subset of cohort IDs to restrict the construction to.

`packageName`
: The name of the package containing the cohort definitions. Can be left NULL if `baseUrl` and `cohortSetReference` have been specified.

`cohortToCreateFile`
: The location of the cohortToCreate file within the package. Is ignored if `baseUrl` and `cohortSetReference` have been specified (i.e. webapi mode takes precedence).

`baseUrl`
: The base URL for the WebApi instance, for example: "http://server.org:80/WebAPI". Can be left NULL if `packageName` and `cohortToCreateFile` have been specified.

`cohortSetReference`
: A data frame with four columns, as described in the details. Can be left NULL if `packageName` and `cohortToCreateFile` have been specified.

`generateInclusionStats`
: Compute and store inclusion rule statistics?

`inclusionStatisticsFolder`
: The folder where the inclusion rule statistics are stored. Can be left NULL if `generateInclusionStats = FALSE`.

`createCohortTable`
: Create the cohort table? If `incremental = TRUE` and the table already exists this will be skipped.

`incremental`
: Create only cohorts that haven't been created before?

`incrementalFolder`
: If `incremental = TRUE`, specify a folder where records are kept of which definition has been executed.

Details

Currently two ways of executing this function are supported, either (1) [Package Mode] embedded in a study package, e.g. by hydrating a the `SkeletonCohortDiagnosticsStudy` package using `Hydra::hydrate` or inserting cohort specifications using `ROhdsiWebApi::insertCohortDefinitionSetInPackage`, or (2) [WebApi Mode] By using a WebApi interface to retrieve the cohort definitions. Note: WebApi mode takes precedence over package mode.

Structure of `cohortSetReference` or `cohortToCreateFile`

cohortId (required) The cohort Id in Atlas for the cohort you want to diagnose.

cohortName (optional) The full name of the cohort. This will be shown in the Shiny app. If not provided, the name used in Atlas will be displayed.

In addition - `cohortToCreateFile` is able to accept additional optional columns

metaData (optional) A JSON with metadata of the cohort that you would like to provided. Logic description may be a metadata object. Other types of metadata objects may include project code, author, version, key words etc.

When using this function in Package Mode: Use the `packageName` and `cohortToCreateFile` to specify the name of the study package, and the name of the cohortToCreate file within that package, respectively

When using this function in WebApi Mode: use the `baseUrl` and `cohortSetReference` to specify how to connect to the WebApi, and which cohorts to fetch, respectively.

The `cohortSetReference` argument maybe a vector of atlas ids or it maybe a data frame object. If `cohortSetReference` is a data frame object:

cohortId (required) The cohort Id in Atlas for the cohort you want to diagnose.

cohortName (optional) The full name of the cohort. This will be shown in the Shiny app. If not provided, the cohort id will be displayed.

Value

A data frame with cohort counts

`launchCohortExplorer` *Launch the CohortExplorer Shiny app*

Description

Launch the CohortExplorer Shiny app

Usage

```
launchCohortExplorer(
  connectionDetails,
  cdmDatabaseSchema,
  cohortDatabaseSchema,
  cohortTable,
  cohortId,
  sampleSize = 100,
  subjectIds = NULL
)
```

Arguments

`connectionDetails`
: An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package.

`cdmDatabaseSchema`
: Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.

`cohortDatabaseSchema`
: Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

cohortTable	Name of the cohort table.
cohortId	The ID of the cohort.
sampleSize	Number of subjects to sample from the cohort. Ignored if subjectIds is specified.
subjectIds	A vector of subject IDs to view.

Details

Launches a Shiny app that allows the user to explore a cohort of interest.

`launchDiagnosticsExplorer` *Launch the Diagnostics Explorer Shiny app*

Description

Launch the Diagnostics Explorer Shiny app

Usage

```
launchDiagnosticsExplorer(
  dataFolder = "data",
  dataFile = "PreMerged.RData",
  connectionDetails = NULL,
  resultsDatabaseSchema = NULL,
  vocabularyDatabaseSchema = NULL,
  vocabularyDatabaseSchemas = resultsDatabaseSchema,
  aboutText = NULL,
  runOverNetwork = FALSE,
  port = 80,
  launch.browser = FALSE
)
```

Arguments

dataFolder	A folder where the premerged file is stored. Use the preMergeDiagnosticsFiles function to generate this file.
dataFile	(Optional) The name of the .RData file with results. It is commonly known as the Premerged file.
connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package, specifying how to connect to the server where the CohortDiagnostics results have been uploaded using the uploadResults function.
resultsDatabaseSchema	The schema on the database server where the CohortDiagnostics results have been uploaded.
vocabularyDatabaseSchema	(Deprecated) Please use vocabularyDatabaseSchemas.

vocabularyDatabaseSchemas
: (optional) A list of one or more schemas on the database server where the vocabulary tables are located. The default value is the value of the resultsDatabaseSchema. We can provide a list of vocabulary schema that might represent different versions of the OMOP vocabulary tables. It allows us to compare the impact of vocabulary changes on Diagnostics.

aboutText
: Text (using HTML markup) that will be displayed in an About tab in the Shiny app. If not provided, no About tab will be shown.

runOverNetwork
: (optional) Do you want the app to run over your network?

port
: (optional) Only used if runOverNetwork = TRUE.

launch.browser
: Should the app be launched in your default browser, or in a Shiny window. Note: copying to clipboard will not work in a Shiny window.

Details

Launches a Shiny app that allows the user to explore the diagnostics

---

preMergeDiagnosticsFiles

Premerge Shiny diagnostics files

Description

This function combines diagnostics results from one or more databases into a single file. The result is a single file that can be used as input for the Diagnostics Explorer Shiny app.

It also checks whether the results conform to the results data model specifications.

Usage

```
preMergeDiagnosticsFiles(dataFolder, tempFolder = tempdir())
```

Arguments

dataFolder
: folder where the exported zip files for the diagnostics are stored. Use the runCohortDiagnostics function to generate these zip files. Zip files containing results from multiple databases may be placed in the same folder.

tempFolder
: A folder on the local file system where the zip files are extracted to. Will be cleaned up when the function is finished. Can be used to specify a temp folder on a drive that has sufficient space if the default system temp space is too limited.

`runCohortCharacterizationDiagnostics` *Get Feature Extraction output for set of cohorts*

Description

Given a set of instantiated cohorts get Characteristics for the cohort using `FeatureExtraction::getDbCovariateData`.

If runTemporalCohortCharacterization argument is TRUE, then the following default covariateSettings object will be created using `RFeatureExtraction::createTemporalCovariateSettings`.

Because of the large file size, the returned object is an `Andromeda::andromeda` class object. Use `CohortDiagnostics::exportFeatureExtractionOutput` to export the characterization results to csv.

Usage

```
runCohortCharacterizationDiagnostics(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  tempEmulationSchema = NULL,
  cohortDatabaseSchema = cdmDatabaseSchema,
  cohortTable = "cohort",
  cohortIds = NULL,
  cdmVersion = 5,
  cutOff = 1e-04,
  covariateSettings = createDefaultCovariateSettings(),
  batchSize = 100
)
```

Arguments

`connectionDetails`
: An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package. Can be left NULL if `connection` is provided.

`connection`
: An object of type `connection` as created using the `connect` function in the DatabaseConnector package. Can be left NULL if `connectionDetails` is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.

`cdmDatabaseSchema`
: Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.

`tempEmulationSchema`
: Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

`cohortDatabaseSchema`
: Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

cohortTable	Name of the cohort table.
cohortIds	Optionally, provide a subset of cohort IDs to restrict the diagnostics to.
cdmVersion	The version of the OMOP CDM. Default 5. (Note: only 5 is supported.)
cutOff	Minimum value of the covariate value, below which data is censored.
covariateSettings	Either an object of type covariateSettings as created using one of the createCovariateSettings (createTemporalCovariateSettings if temporal characterization) function in the FeatureExtraction package, or a list of such objects. If unspecified, default covariate settings as specified by FeatureExtraction is computed, this is sufficient for presenting default table 1. See documentation of FeatureExtraction on how to specify CovariateSettings object.
batchSize	Optional, default set to 100 If running characterization on larget set of cohorts, this function allows you to batch them into chunks that run as a batch.

`runCohortDiagnostics` *Run cohort diagnostics*

Description

Runs cohort diagnostics on cohorts specified in cohortToCreateFile (file) or cohortSetReference. The function checks if the specified cohorts are instantiated, and only runs diagnostics on the instantiated cohorts (i.e. > 0 rows in cohort table).

Characterization: If runTemporalCohortCharacterization argument is TRUE, then `RFeatureExtraction::createTempo` is used as default.

Usage

```
runCohortDiagnostics(
  packageName = NULL,
  cohortToCreateFile = "settings/CohortsToCreate.csv",
  baseUrl = NULL,
  cohortSetReference = NULL,
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  tempEmulationSchema = NULL,
  cohortDatabaseSchema,
  vocabularyDatabaseSchema = cdmDatabaseSchema,
  cohortTable = "cohort",
  cohortIds = NULL,
  inclusionStatisticsFolder = file.path(exportFolder, "inclusionStatistics"),
  exportFolder,
  databaseId,
  databaseName = databaseId,
  databaseDescription = databaseId,
  cdmVersion = 5,
  runInclusionStatistics = TRUE,
  runConceptSetDiagnostics = TRUE,
```

```
  runIncludedSourceConcepts = FALSE,
  runOrphanConcepts = FALSE,
  runVisitContext = TRUE,
  runBreakdownIndexEvents = FALSE,
  runIncidenceRate = TRUE,
  runCohortTimeSeries = TRUE,
  runDataSourceTimeSeries = TRUE,
  runCohortRelationship = TRUE,
  runCohortCharacterization = TRUE,
  covariateSettings = list(FeatureExtraction::createDefaultCovariateSettings(),
    FeatureExtraction::createCovariateSettings(useVisitCountLongTerm = TRUE,
    useVisitCountShortTerm = TRUE, useVisitConceptCountLongTerm = TRUE,
   useVisitConceptCountShortTerm = TRUE, useDemographicsPriorObservationTime = TRUE,
   useDemographicsPostObservationTime = TRUE, useDemographicsTimeInCohort = TRUE,
    useDemographicsIndexYearMonth = TRUE, )),
  runTemporalCohortCharacterization = TRUE,

   temporalCovariateSettings = FeatureExtraction::createTemporalCovariateSettings(useConditionOcc
   = TRUE, useDrugEraStart = TRUE, useProcedureOccurrence = TRUE, useMeasurement = TRUE,
    temporalStartDays = c(-365, -30, 0, 1, 31, seq(from = -421, to = -31, by = 30),
   seq(from = 0, to = 390, by = 30)), temporalEndDays = c(-31, -1, 0, 30, 365, seq(from
    = -391, to = -1, by = 30), seq(from = 30, to = 420, by = 30))),
  minCellCount = 5,
  incremental = FALSE,
  incrementalFolder = file.path(exportFolder, "incremental")
)
```

Arguments

packageName	The name of the package containing the cohort definitions. Can be left NULL if baseUrl and cohortSetReference have been specified.
cohortToCreateFile	The location of the cohortToCreate file within the package. Is ignored if baseUrl and cohortSetReference have been specified (i.e. webapi mode takes precedence).
baseUrl	The base URL for the WebApi instance, for example: "http://server.org:80/WebAPI". Can be left NULL if packageName and cohortToCreateFile have been specified.
cohortSetReference	A data frame with four columns, as described in the details. Can be left NULL if packageName and cohortToCreateFile have been specified.
connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
cdmDatabaseSchema	Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema

name, for example 'cdm_data.dbo'.

`oracleTempSchema`
: DEPRECATED by DatabaseConnector: use `tempEmulationSchema` instead.

`tempEmulationSchema`
: Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

`cohortDatabaseSchema`
: Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

`vocabularyDatabaseSchema`
: Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.

`cohortTable`
: Name of the cohort table.

`cohortIds`
: Optionally, provide a subset of cohort IDs to restrict the diagnostics to.

`inclusionStatisticsFolder`
: The folder where the inclusion rule statistics are stored. Can be left NULL if `runInclusionStatistics = FALSE`.

`exportFolder`
: The folder where the output will be exported to. If this folder does not exist it will be created.

`databaseId`
: A short string for identifying the database (e.g. 'Synpuf').

`databaseName`
: The full name of the database. If NULL, defaults to databaseId.

`databaseDescription`
: A short description (several sentences) of the database. If NULL, defaults to databaseId.

`cdmVersion`
: The version of the OMOP CDM. Default 5. (Note: only 5 is supported.)

`runInclusionStatistics`
: Generate and export statistic on the cohort inclusion rules?

`runConceptSetDiagnostics`
: Concept Set Diagnostics includes concept counts, concepts in data source, index event breakdown, concept cooccurrence, excluded concepts, resolved concepts. This function call now supersedes runIncludedSourceConcepts, runOrphanConcepts, runBreakdownIndexEvents.

`runIncludedSourceConcepts`
: (Deprecated) Generate and export the source concepts included in the cohorts?

`runOrphanConcepts`
: (Deprecated) Generate and export potential orphan concepts?

`runVisitContext`
: Generate and export index-date visit context?

`runBreakdownIndexEvents`
: (Deprecated) Generate and export the breakdown of index events?

`runIncidenceRate`
: Generate and export the cohort incidence rates?

`runCohortTimeSeries`
: Generate and export the cohort level time series?

`runDataSourceTimeSeries`
: Generate and export the Data source level time series? i.e. using all persons found in observation period table.

`runCohortRelationship`
: Do you want to compute temporal relationship between the cohorts being diagnosed. This diagnostics is needed for cohort as feature characterization.

`runCohortCharacterization`
: Generate and export the cohort characterization? Only records with values greater than 0.0001 are returned.

`covariateSettings`
: Either an object of type `covariateSettings` as created using one of the createCovariateSettings function in the FeatureExtraction package, or a list of such objects.

`runTemporalCohortCharacterization`
: Generate and export the temporal cohort characterization? Only records with values greater than 0.001 are returned.

`temporalCovariateSettings`
: Either an object of type `covariateSettings` as created using one of the createTemporalCovariateSettings function in the FeatureExtraction package, or a list of such objects.

`minCellCount`
: The minimum cell count for fields contains person counts or fractions.

`incremental`
: Create only cohort diagnostics that haven't been created before?

`incrementalFolder`
: If `incremental = TRUE`, specify a folder where records are kept of which cohort diagnostics has been executed.

Details

Currently two ways of executing this function are supported, either (1) [Package Mode] embedded in a study package, e.g. by hydrating a the `SkeletonCohortDiagnosticsStudy` package using `Hydra::hydrate` or inserting cohort specifications using `ROhdsiWebApi::insertCohortDefinitionSetInPackage`, or (2) [WebApi Mode] By using a WebApi interface to retrieve the cohort definitions. Note: WebApi mode takes precedence over package mode.

Structure of `cohortSetReference` or `cohortToCreateFile`

cohortId (required) The cohort Id in Atlas for the cohort you want to diagnose.

cohortName (optional) The full name of the cohort. This will be shown in the Shiny app. If not provided, the name used in Atlas will be displayed.

In addition - `cohortToCreateFile` is able to accept additional optional columns

metaData (optional) A JSON with metadata of the cohort that you would like to provided. Logic description may be a metadata object. Other types of metadata objects may include project code, author, version, key words etc.

When using this function in Package Mode: Use the `packageName` and `cohortToCreateFile` to specify the name of the study package, and the name of the cohortToCreate file within that package, respectively

When using this function in WebApi Mode: use the `baseUrl` and `cohortSetReference` to specify how to connect to the WebApi, and which cohorts to fetch, respectively.

The `cohortSetReference` argument maybe a vector of atlas ids or it maybe a data frame object. If `cohortSetReference` is a data frame object:

cohortId (required) The cohort Id in Atlas for the cohort you want to diagnose.

cohortName (optional) The full name of the cohort. This will be shown in the Shiny app. If not provided, the cohort id will be displayed.

`runCohortRelationshipDiagnostics` *Given a set of cohorts get relationships between the cohorts.*

Description

Given a set of cohorts, get temporal relationships between the cohort_start_date of the cohorts.

Usage

```
runCohortRelationshipDiagnostics(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema,
  tempEmulationSchema = NULL,
  cohortTable = "cohort",
  targetCohortIds,
  comparatorCohortIds
)
```

Arguments

`connectionDetails`	An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package. Can be left NULL if `connection` is provided.
`connection`	An object of type `connection` as created using the `connect` function in the DatabaseConnector package. Can be left NULL if `connectionDetails` is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
`cohortDatabaseSchema`	Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
`tempEmulationSchema`	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.
`cohortTable`	Name of the cohort table.
`targetCohortIds`	A vector of one or more Cohort Ids for use as target cohorts.
`comparatorCohortIds`	A vector of one or more Cohort Ids for use as feature/comparator cohorts.

`runCohortTimeSeriesDiagnostics` *Given a set of instantiated cohorts get time series for the cohorts.*

Description

This function first generates a calendar period table, that has calendar intervals between the `timeSeriesMinDate` and `timeSeriesMaxDate`. Calendar Month, Quarter and year are supported. For each of the calendar interval, time series data are computed. The returned object is a R dataframe that will need to be converted to a time series object to perform time series analysis.

Data Source time series: computes time series at the data source level i.e. observation period table. This output is NOT limited to individuals in the cohort table but is for ALL people in the datasource (i.e. present in observation period table)

Usage

```
runCohortTimeSeriesDiagnostics(
  connectionDetails = NULL,
  connection = NULL,
  tempEmulationSchema = NULL,
  cdmDatabaseSchema,
  cohortDatabaseSchema = cdmDatabaseSchema,
  cohortTable = "cohort",
  runCohortTimeSeries = TRUE,
  runDataSourceTimeSeries = TRUE,
  timeSeriesMinDate = as.Date("1980-01-01"),
  timeSeriesMaxDate = as.Date(Sys.Date()),
  cohortIds = NULL
)
```

Arguments

`connectionDetails`
: An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package. Can be left NULL if `connection` is provided.

`connection`
: An object of type `connection` as created using the `connect` function in the DatabaseConnector package. Can be left NULL if `connectionDetails` is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.

`tempEmulationSchema`
: Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

`cdmDatabaseSchema`
: Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.

`cohortDatabaseSchema`
: Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

cohortTable Name of the cohort table.

runCohortTimeSeries
Generate and export the cohort level time series?

runDataSourceTimeSeries
Generate and export the Data source level time series? i.e. using all persons found in observation period table.

timeSeriesMinDate
(optional) Minimum date for time series. Default value January 1st 1980.

timeSeriesMaxDate
(optional) Maximum date for time series. Default value System date.

cohortIds A vector of one or more Cohort Ids to compute time distribution for.

---

runConceptSetDiagnostics *Run concept set diagnostics*

Description

Runs concept set diagnostics on a set of cohorts. For index event breakdown, the cohorts need to be instantiated.

Usage

```
runConceptSetDiagnostics(
  connection = NULL,
  connectionDetails = NULL,
  tempEmulationSchema = NULL,
  cdmDatabaseSchema,
  vocabularyDatabaseSchema = cdmDatabaseSchema,
  cohorts,
  cohortIds = NULL,
  cohortDatabaseSchema = NULL,
  keepCustomConceptId = FALSE,
  cohortTable = NULL
)
```

Arguments

connection An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.

connectionDetails
An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.

tempEmulationSchema
Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

`cdmDatabaseSchema`
: Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.

`vocabularyDatabaseSchema`
: Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.

`cohorts`
: A dataframe object with required fields cohortId, sql, json, cohortName

`cohortIds`
: Optionally, provide a subset of cohort IDs to restrict the diagnostics to.

`cohortDatabaseSchema`
: Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

`keepCustomConceptId`
: (Optional) Default FALSE. Do you want to keep concept id above 2 billion. Per OMOP conventions any conceptId >= 2 billion are considered site specific custom value that are not shipped as part of default OMOP vocabulary tables.

`cohortTable`
: Name of the cohort table.

`runIncidenceRateDiagnostics` *Given a set of instantiated cohorts get Incidence Rate for the cohorts.*

Description

This function computes incidence rate, one cohort at a time.

Usage

```
runIncidenceRateDiagnostics(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema,
  cohortTable,
  cdmDatabaseSchema,
  vocabularyDatabaseSchema = cdmDatabaseSchema,
  cdmVersion = 5,
  tempEmulationSchema = tempEmulationSchema,
  firstOccurrenceOnly = TRUE,
  washoutPeriod = 365,
  cohortId
)
```

Arguments

`connectionDetails`
: An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package. Can be left NULL if `connection` is provided.

connection An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.

cohortDatabaseSchema
Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

cohortTable Name of the cohort table.

cdmDatabaseSchema
Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.

vocabularyDatabaseSchema
Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.

cdmVersion Only CDM version 5 is supported.

tempEmulationSchema
Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

firstOccurrenceOnly
Compute for first occurrence of subject in the cohort.

washoutPeriod (Optional) Washout period to use. The default value is either 365 days or minimum prior observation period requirement specified in cohort definition.

cohortId A Cohort Id to compute time distribution for.

---

runVisitContextDiagnostics
Given a set of instantiated cohorts get the visit context in relation to cohort start date.

Description

This function returns the types of visits experienced by persons in the cohort in relation to cohort start date.

Usage

```
runVisitContextDiagnostics(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  tempEmulationSchema = NULL,
  cohortDatabaseSchema,
  vocabularyDatabaseSchema,
  cohortTable = "cohort",
  cohortIds,
  cdmVersion = 5
)
```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
cdmDatabaseSchema	Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.
tempEmulationSchema	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.
cohortDatabaseSchema	Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
vocabularyDatabaseSchema	Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.
cohortTable	Name of the cohort table.
cohortIds	A vector of one or more Cohort Ids to compute visit context for.
cdmVersion	The version of the OMOP CDM. Default 5. (Note: only 5 is supported.)

---

takepackageDependencySnapshot

Take a snapshot of the R environment

Description

Take a snapshot of the R environment

Usage

```
takepackageDependencySnapshot()
```

Details

This function records all versions used in the R environment as used by runCohortDiagnostics. This function was borrowed from OhdsiRTools

Value

A data frame listing all the dependencies of the root package and their version numbers, in the order in which they should be installed.

uploadResults *Upload results to the database server.*

Description

Requires the results data model tables have been created using the `createResultsDataModel` function.

Set the POSTGRES_PATH environmental variable to the path to the folder containing the psql executable to enable bulk upload (recommended).

Usage

```
uploadResults(
  connectionDetails = NULL,
  schema,
  zipFileName,
  forceOverWriteOfSpecifications = FALSE,
  purgeSiteDataBeforeUploading = TRUE,
  tempFolder = tempdir()
)
```

Arguments

`connectionDetails`	An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package.
`schema`	The schema on the postgres server where the tables have been created.
`zipFileName`	The name of the zip file.
`forceOverWriteOfSpecifications`	If TRUE, specifications of the phenotypes, cohort definitions, and analysis will be overwritten if they already exist on the database. Only use this if these specifications have changed since the last upload.
`purgeSiteDataBeforeUploading`	If TRUE, before inserting data for a specific databaseId all the data for that site will be dropped. This assumes the input zip file contains the full data for that data site.
`tempFolder`	A folder on the local file system where the zip files are extracted to. Will be cleaned up when the function is finished. Can be used to specify a temp folder on a drive that has sufficient space if the default system temp space is too limited.

Index