Package 'CohortDiagnostics'

July 31, 2020

Type Package

Title Diagnostics for OHDSI Studies

Version 1.1.0

Date 2020-07-30

Maintainer Gowtham Rao <gowthamrao@gmail.com>

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

Depends DatabaseConnector (¿= 3.0.0),
R (¿= 3.5.0)

Imports Andromeda,
checkmate,
digest,
dplyr (¿= 1.0.0),
FeatureExtraction (¿= 3.0.1),
ggplot2,
ParallelLogger (¿= 2.0.0),
readr,
rlang,
RJSONIO,
ROhdsiWebApi (¿= 1.0.0),
SqlRender (¿= 1.6.7),
stringr,
tibble (¿= 3.0.0),
tidyr (¿= 1.0.0)

Suggests DT,
Eunomia,
RSQLite (¿ 2.2.0),
htmltools,
knitr,
plotly,
RColorBrewer,
rmarkdown,
shiny,
shinydashboard,
VennDiagram,
testthat

Remotes ohdsi/Eunomia,

ohdsi/FeatureExtraction,
ohdsi/ROhdsiWebApi,
ohdsi/SqlRender,
r-dbi/RSQLite

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:

`breakDownIndexEvents` *Break down index events*

Description

For the concepts included in the index event definition, count how often they are encountered at the cohort index date.

Usage

```
breakDownIndexEvents(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  cohortDatabaseSchema = cdmDatabaseSchema,
  cohortTable = "cohort",
  baseUrl = NULL,
  webApiCohortId = NULL,
  cohortJson = NULL,
  cohortSql = NULL,
  cohortId = 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.
`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`	Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.
`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.
`baseUrl`	The base URL for the WebApi instance, for example: "http://server.org:80/WebAPI". Needn't be provided if `cohortJson` and `cohortSql` are provided.
`webApiCohortId`	The ID of the cohort in the WebAPI instance. Needn't be provided if `cohortJson` and `cohortSql` are provided.

`cohortJson`	A character string containing the JSON of a cohort definition. Needn't be provided if `baseUrl` and `cohortId` are provided.
`cohortSql`	The OHDSI SQL representation of the same cohort definition. Needn't be provided if `baseUrl` and `cohortId` are provided.
`cohortId`	The cohort definition ID used to reference the cohort in the cohort table.

Value

A data frame with concepts, and per concept the count of how often the concept was encountered at the index date.

`compareCohortCharacteristics` *Compare cohort characteristics*

Description

Compare the characteristics of two cohorts, computing the standardized difference of the mean.

Usage

```
compareCohortCharacteristics(characteristics1, characteristics2)
```

Arguments

`characteristics1`	Characteristics of the first cohort, as created using the `getCohortCharacteristics` function.
`characteristics2`	Characteristics of the second cohort, as created using the `getCohortCharacteristics` function.

Value

A data frame comparing the characteristics of the two cohorts.

`computeCohortOverlap` *Compute overlap between two cohorts*

Description

Computes the overlap between a target and a comparator cohort.

Usage

```
computeCohortOverlap(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema,
  cohortTable = "cohort",
  targetCohortId,
  comparatorCohortId
)
```

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.
targetCohortId	The cohort definition ID used to reference the target cohort in the cohort table.
comparatorCohortId	The cohort definition ID used to reference the comparator cohort in the cohort table.

Value

A data frame with overlap statistics.

---

createCohortTable *Create cohort table(s)*

Description

This function creates an empty cohort table. Optionally, additional empty tables are created to store statistics on the various inclusion criteria.

Usage

```
createCohortTable(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema,
  cohortTable = "cohort",
```

```
  createInclusionStatsTables = FALSE,
  resultsDatabaseSchema = cohortDatabaseSchema,
  cohortInclusionTable = paste0(cohortTable, "_inclusion"),
  cohortInclusionResultTable = paste0(cohortTable, "_inclusion_result"),
  cohortInclusionStatsTable = paste0(cohortTable, "_inclusion_stats"),
  cohortSummaryStatsTable = paste0(cohortTable, "_summary_stats")
)
```

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.

`createInclusionStatsTables`
: Create the four additional tables for storing inclusion rule statistics?

`resultsDatabaseSchema`
: Schema name where the statistics tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

`cohortInclusionTable`
: Name of the inclusion table, one of the tables for storing inclusion rule statistics.

`cohortInclusionResultTable`
: Name of the inclusion result table, one of the tables for storing inclusion rule statistics.

`cohortInclusionStatsTable`
: Name of the inclusion stats table, one of the tables for storing inclusion rule statistics.

`cohortSummaryStatsTable`
: Name of the summary stats table, one of the tables for storing inclusion rule statistics.

`createConceptCountsTable` *Create concept counts table*

Description

Create a table with counts of how often each concept ID occurs in the CDM.

Usage

```
createConceptCountsTable(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  conceptCountsDatabaseSchema = cdmDatabaseSchema,
  conceptCountsTable = "concept_counts",
  conceptCountsTableIsTemp = FALSE
)
```

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'.
`oracleTempSchema`	Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.
`conceptCountsDatabaseSchema`	Schema name where your concept counts table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'. Ignored if `conceptCountsTableIsTemp = TRUE`.
`conceptCountsTable`	Name of the concept counts table. This table can be created using the `createConceptCountsTable`.
`conceptCountsTableIsTemp`	Is the concept counts table a temp table?

`findCohortIncludedSourceConcepts` *Check source codes used in a cohort definition*

Description

This function first extracts all concept sets used in a cohort definition. Then, for each concept set the concept found in the CDM database the contributing source codes are identified.

Usage

```
findCohortIncludedSourceConcepts(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  baseUrl = NULL,
  webApiCohortId = NULL,
  cohortJson = NULL,
  cohortSql = NULL,
  byMonth = FALSE,
  useSourceValues = FALSE
)
```

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'.
oracleTempSchema	Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.
baseUrl	The base URL for the WebApi instance, for example: "http://server.org:80/WebAPI". Needn't be provided if cohortJson and cohortSql are provided.
webApiCohortId	The ID of the cohort in the WebAPI instance. Needn't be provided if cohortJson and cohortSql are provided.
cohortJson	A character string containing the JSON of a cohort definition. Needn't be provided if baseUrl and cohortId are provided.
cohortSql	The OHDSI SQL representation of the same cohort definition. Needn't be provided if baseUrl and cohortId are provided.
byMonth	Compute counts by month? If FALSE, only overall counts are computed.
useSourceValues	Use the source_value fields to find the codes used in the data? If not, this analysis will rely entirely on the source_concept_id fields instead. Note that, depending on the source data and ETL, it might be possible for the source_value fields to contain patient-identifiable information by accident.

Value

A data frame with source codes, with counts per domain how often the code was encountered in the CDM.

`findCohortOrphanConcepts` *Find orphan concepts for all concept sets in a cohort*

Description

Searches for concepts that should belong to the concept sets in a cohort definition but don't, for example because of missing source-to-standard concept maps, or erroneous hierarchical relationships.

Usage

```
findCohortOrphanConcepts(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  baseUrl = NULL,
  webApiCohortId = NULL,
  cohortJson = NULL,
  conceptCountsDatabaseSchema = cdmDatabaseSchema,
  conceptCountsTable = "concept_counts",
  conceptCountsTableIsTemp = FALSE
)
```

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'.
`oracleTempSchema`	Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.
`baseUrl`	The base URL for the WebApi instance, for example: "http://server.org:80/WebAPI". Needn't be provided if `cohortJson` is provided.
`webApiCohortId`	The ID of the cohort in the WebAPI instance. Needn't be provided if `cohortJson` is provided.
`cohortJson`	A character string containing the JSON of a cohort definition. Needn't be provided if `baseUrl` and `webApiCohortId` are provided.

`conceptCountsDatabaseSchema`
: Schema name where your concept counts table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'. Ignored if `conceptCountsTableIsTemp = TRUE`.

`conceptCountsTable`
: Name of the concept counts table. This table can be created using the `createConceptCountsTable`.

`conceptCountsTableIsTemp`
: Is the concept counts table a temp table?

Details

Logically, this function performs the following steps for each concept set expression in the cohort definition:

- Given the concept set expression, find all included concepts.
- Find all names of the input concepts, including synonyms, and the names of source concepts that map to them.
- Search for concepts (standard and source) that contain any of those names as substring.
- Filter those concepts to those that are not in the original set of concepts (i.e. orphans).
- Restrict the set of orphan concepts to those that appear in the CDM database and across network concept prevalence (as either source concept or standard concept).

Value

A data frame with orphan concepts, with counts how often the code was encountered in the CDM.

`findOrphanConcepts` *Find (source) concepts that do not roll up to their ancestor(s)*

Description

Searches for concepts that should belong to the set of concepts but don't, for example because of missing source-to-standard concept maps, or erroneous hierarchical relationships.

Usage

```
findOrphanConcepts(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  conceptIds,
  conceptCountsDatabaseSchema = cdmDatabaseSchema,
  conceptCountsTable = "concept_counts",
  conceptCountsTableIsTemp = FALSE
)
```

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'.

`oracleTempSchema`
: Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.

`conceptIds`
: A vector of concept IDs for which we want to find orphans.

`conceptCountsDatabaseSchema`
: Schema name where your concept counts table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'. Ignored if `conceptCountsTableIsTemp = TRUE`.

`conceptCountsTable`
: Name of the concept counts table. This table can be created using the `createConceptCountsTable`.

`conceptCountsTableIsTemp`
: Is the concept counts table a temp table?

Details

Logically, this function performs the following steps for the input set of concept IDs:

- Find all names of the input concepts, including synonyms, and the names of source concepts that map to them.
- Search for concepts (standard and source) that contain any of those names as substring.
- Filter those concepts to those that are not in the original set of concepts (i.e. orphans).
- Restrict the set of orphan concepts to those that appear in the CDM database and across network concept prevalence (as either source concept or standard concept).

Value

A data frame with orphan concepts, with counts how often the code was encountered in the CDM.

`getCohortCharacteristics` *Create characterization of a cohort*

Description

Computes features using all drugs, conditions, procedures, etc. observed on or prior to the cohort index date.

Usage

```
getCohortCharacteristics(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  cohortDatabaseSchema = cdmDatabaseSchema,
  cohortTable = "cohort",
  cohortId,
  covariateSettings
)
```

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'.

`oracleTempSchema`
: Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.

`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 cohort definition ID used to reference the cohort in the cohort table.

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

Value

A data frame with cohort characteristics.

`getCohortCounts` *Count 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 definition ID(s0 used to reference the cohort in the cohort table. If left empty, all cohorts in the table will be included.

Value

A data frame with cohort counts

`getCohortsJsonAndSql` *Get cohorts JSON and parameterized OHDSI SQL*

Description

This function may be used to collect a cohorts JSON and OHDSI SQL. Based on whether a baseUrl is available, the function will collect the specifications from either from WebApi or a Package.

Usage

```
getCohortsJsonAndSql(
  packageName = NULL,
  cohortToCreateFile = "settings/CohortsToCreate.csv",
  baseUrl = NULL,
  cohortSetReference = NULL,
  cohortIds = NULL
)
```

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. The cohortToCreateFile must be .csv file that is expected to be read into a dataframe object identical to requirements for `cohortSetReference` argument.
`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.
`cohortIds`	Optionally, provide a subset of cohort IDs to restrict the diagnostics to.

Details

Currently two ways of executing this function are supported, either (1) [Package Mode] embedded in a study package, assuming the cohort definitions are stored in that package using the `ROhdsiWebApi::insertCohortDefinitionSetInPackage`, or (2) [WebApi Mode] By using a WebApi interface to retrieve the cohort definitions.

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.

Note: if the parameters for both Package Mode and WebApi Mode are provided, then Package mode is preferred.

The `cohortSetReference` argument must be a data frame with the following columns:

atlasId The cohort ID in ATLAS.

atlasName The full name of the cohort. This will be shown in the Shiny app.

cohortId The cohort ID to use in the package. Usually the same as the cohort ID in ATLAS.

name A short name for the cohort, to use to create file names. do not use special characters.

Value

The function will return a R list object with cohort information including specifications such as JSON and SQL.

Examples

```
## Not run:
cohorts <- getCohortsJsonAndSql(packageName = 'cohortDiagnostics',
                                baseUrl = "http://server.org:80/WebAPI")

## End(Not run)
```

---

getIncidenceRate *Compute incidence rate for a cohort*

Description

Returns yearly incidence rate stratified by age and gender

Usage

```
getIncidenceRate(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema,
  cohortTable,
  cdmDatabaseSchema,
  oracleTempSchema = oracleTempSchema,
  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'.

`oracleTempSchema`
: Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.

`firstOccurrenceOnly`
: Use only the first occurrence of the cohort per person?

`washoutPeriod`
: The minimum amount of observation time required before the occurrence of a cohort entry. This is also used to eliminate immortal time from the denominator.

`cohortId`
: The cohort definition ID used to reference the cohort in the cohort table.

Value

Returns a data frame of cohort count, person year count, and incidence rate per 1000 persons years with the following stratifications: 1) no stratification, 2) gender stratification, 3) age (10-year) stratification, 4) calendar year and age (10-year) stratification, 5) calendar year and gender stratification, 6) calendar year, age (10-year), and gender stratification with option to save dataframes.

`getInclusionStatistics` *Get statistics on cohort inclusion criteria*

Description

Get statistics on cohort inclusion criteria

Usage

```
getInclusionStatistics(
  connectionDetails = NULL,
  connection = NULL,
  resultsDatabaseSchema,
  cohortId,
  simplify = TRUE,
  cohortTable = "cohort",
  cohortInclusionTable = paste0(cohortTable, "_inclusion"),
  cohortInclusionResultTable = paste0(cohortTable, "_inclusion_result"),
  cohortInclusionStatsTable = paste0(cohortTable, "_inclusion_stats"),
  cohortSummaryStatsTable = paste0(cohortTable, "_summary_stats")
)
```

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.

`resultsDatabaseSchema`
: Schema name where the statistics tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

`cohortId`
: The cohort definition ID used to reference the cohort in the cohort table.

`simplify`
: Simply output the attrition table?

`cohortTable`
: Name of the cohort table. Used only to conveniently derive names of the four rule statistics tables.

`cohortInclusionTable`
: Name of the inclusion table, one of the tables for storing inclusion rule statistics.

`cohortInclusionResultTable`
: Name of the inclusion result table, one of the tables for storing inclusion rule statistics.

`cohortInclusionStatsTable`
: Name of the inclusion stats table, one of the tables for storing inclusion rule statistics.

`cohortSummaryStatsTable`
: Name of the summary stats table, one of the tables for storing inclusion rule statistics.

Value

If 'simplify = TRUE', this function returns a single data frame. Else a list of data frames is returned.

`getInclusionStatisticsFromFiles` *Get inclusion criteria statistics from files*

Description

Gets inclusion criteria statistics from files, as stored when using the `ROhdsiWebApi::insertCohortDefinitionSe` function with `generateStats = TRUE`.

Usage

```
getInclusionStatisticsFromFiles(
  cohortId,
  folder,
  cohortInclusionFile = file.path(folder, "cohortInclusion.csv"),
  cohortInclusionResultFile = file.path(folder, "cohortIncResult.csv"),
  cohortInclusionStatsFile = file.path(folder, "cohortIncStats.csv"),
  cohortSummaryStatsFile = file.path(folder, "cohortSummaryStats.csv"),
  simplify = TRUE
)
```

Arguments

cohortId	The cohort definition ID used to reference the cohort in the cohort table.
folder	The path to the folder where the inclusion statistics are stored.
cohortInclusionFile	Name of the inclusion table, one of the tables for storing inclusion rule statistics.
cohortInclusionResultFile	Name of the inclusion result table, one of the tables for storing inclusion rule statistics.
cohortInclusionStatsFile	Name of the inclusion stats table, one of the tables for storing inclusion rule statistics.
cohortSummaryStatsFile	Name of the summary stats table, one of the tables for storing inclusion rule statistics.
simplify	Simply output the attrition table?

Value

If 'simplify = TRUE', this function returns a single data frame. Else a list of data frames is returned.

---

getTimeDistributions *Get time distributions of a cohort*

Description

Computes the distribution of the observation time before and after index, and time within a cohort.

Usage

```
getTimeDistributions(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  cohortDatabaseSchema = cdmDatabaseSchema,
```

```
  cohortTable = "cohort",
  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.
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	Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.
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 cohort definition ID used to reference the cohort in the cohort table.

Value

A data frame with time distributions

---

instantiateCohort *Instantiate a cohort*

Description

This function instantiates the cohort in the cohort table. Optionally, the inclusion rule statistics are computed and stored in the inclusion rule statistics tables described in createCohortTable).

Usage

```
instantiateCohort(
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  cohortDatabaseSchema = cdmDatabaseSchema,
  cohortTable = "cohort",
  baseUrl = NULL,
```

```
  webApiCohortId = NULL,
  cohortJson = NULL,
  cohortSql = NULL,
  cohortId = NULL,
  generateInclusionStats = FALSE,
  resultsDatabaseSchema = cohortDatabaseSchema,
  cohortInclusionTable = paste0(cohortTable, "_inclusion"),
  cohortInclusionResultTable = paste0(cohortTable, "_inclusion_result"),
  cohortInclusionStatsTable = paste0(cohortTable, "_inclusion_stats"),
  cohortSummaryStatsTable = paste0(cohortTable, "_summary_stats")
)
```

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'.
`oracleTempSchema`	Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.
`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.
`baseUrl`	The base URL for the WebApi instance, for example: "http://server.org:80/WebAPI". Needn't be provided if `cohortJson` and `cohortSql` are provided.
`webApiCohortId`	The ID of the cohort in the WebAPI instance. Needn't be provided if `cohortJson` and `cohortSql` are provided.
`cohortJson`	A character string containing the JSON of a cohort definition. Needn't be provided if `baseUrl` and `cohortId` are provided.
`cohortSql`	The OHDSI SQL representation of the same cohort definition. Needn't be provided if `baseUrl` and `cohortId` are provided.
`cohortId`	The cohort definition ID used to reference the cohort in the cohort table.
`generateInclusionStats`	Compute and store inclusion rule statistics?
`resultsDatabaseSchema`	Schema name where the statistics tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

`cohortInclusionTable`
: Name of the inclusion table, one of the tables for storing inclusion rule statistics.

`cohortInclusionResultTable`
: Name of the inclusion result table, one of the tables for storing inclusion rule statistics.

`cohortInclusionStatsTable`
: Name of the inclusion stats table, one of the tables for storing inclusion rule statistics.

`cohortSummaryStatsTable`
: Name of the summary stats table, one of the tables for storing inclusion rule statistics.

`instantiateCohortSet` *Instantiate a set of cohort*

Description

This function instantiates a set of cohort in the 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,
  oracleTempSchema = NULL,
  cohortDatabaseSchema = cdmDatabaseSchema,
  cohortTable = "cohort",
  cohortIds = NULL,
  packageName = NULL,
  cohortToCreateFile = "settings/CohortsToCreate.csv",
  baseUrl = NULL,
  cohortSetReference = NULL,
  generateInclusionStats = FALSE,
  inclusionStatisticsFolder = NULL,
  createCohortTable = FALSE,
  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'.

`oracleTempSchema`
Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.

`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. The cohortToCreateFile must be .csv file that is expected to be read into a dataframe object identical to requirements for `cohortSetReference` argument.

`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, assuming the cohort definitions are stored in that package using the `ROhdsiWebApi::insertCohortDefinitionSetInPackage`, or (2) [WebApi Mode] By using a WebApi interface to retrieve the cohort definitions.

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.

Note: if the parameters for both Package Mode and WebApi Mode are provided, then Package mode is preferred.

The `cohortSetReference` argument must be a data frame with the following columns:

atlasId The cohort ID in ATLAS.

atlasName The full name of the cohort. This will be shown in the Shiny app.

cohortId The cohort ID to use in the package. Usually the same as the cohort ID in ATLAS.

name A short name for the cohort, to use to create file names. do not use special characters.

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, launch.browser = FALSE)
```

Arguments

dataFolder	A 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 can be placed in the same folder.
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

---

plotincidenceRate *Plot incidence rate by year, age, and/or gender*

Description

Characterizes the incidence rate of a cohort definition.

Usage

```
plotincidenceRate(
  incidenceRate,
  minPersonYears = 1000,
  stratifyByAge = TRUE,
  stratifyByGender = TRUE,
  stratifyByCalendarYear = TRUE,
  fileName = NULL
)
```

Arguments

`incidenceRate`	Incidence rate time series data for plotting generated using `getIncidenceRate` function.
`minPersonYears`	Estimates get very unstable with low background counts, so removing them makes for cleaner plots.
`stratifyByAge`	Should the plot be stratified by age?
`stratifyByGender`	Should the plot be stratified by gender?
`stratifyByCalendarYear`	Should the plot be stratified by calendar year?
`fileName`	Optional: name of the file where the plot should be saved, for example 'plot.png'. See the function `ggsave` in the ggplot2 package for supported file formats.

Details

Generates time series plots of the incidence rate per 1000 person years of cohort entry by year, age, and/or gender.

Value

A ggplot object. Use the `ggsave` function to save to file in a different format.

`preMergeDiagnosticsFiles` *Premerge Shiny diagnostics files*

Description

If there are many diagnostics files, starting the Shiny app may take a very long time. This function already does most of the preprocessing, increasing loading speed.

The merged data will be stored in the same folder, and will automatically be recognized by the Shiny app.

Usage

```
preMergeDiagnosticsFiles(dataFolder)
```

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 can be placed in the same folder.

runCohortDiagnostics *Run cohort diagnostics*

Description

Runs the cohort diagnostics on all (or a subset of) the cohorts instantiated using the ROhdsiWebApi::insertCohortDefinitionSetInPackage function. Assumes the cohorts have already been instantiated.

Characterization: If runTemporalCohortCharacterization argument is TRUE, then the following default covariateSettings object will be created using RFeatureExtraction::createTemporalCovariateS
Alternatively, a covariate setting object may be created using the above as an example.

Usage

```
runCohortDiagnostics(
  packageName = NULL,
  cohortToCreateFile = "settings/CohortsToCreate.csv",
  baseUrl = NULL,
  cohortSetReference = NULL,
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  cohortDatabaseSchema,
  cohortTable = "cohort",
  cohortIds = NULL,
  inclusionStatisticsFolder = NULL,
  exportFolder,
  databaseId,
  databaseName = databaseId,
  databaseDescription = "",
  runInclusionStatistics = TRUE,
  runIncludedSourceConcepts = TRUE,
  runOrphanConcepts = TRUE,
  runTimeDistributions = TRUE,
  runBreakdownIndexEvents = TRUE,
  runIncidenceRate = TRUE,
  runCohortOverlap = TRUE,
  runCohortCharacterization = TRUE,
  covariateSettings = FeatureExtraction::createDefaultCovariateSettings(),
  runTemporalCohortCharacterization = TRUE,

   temporalCovariateSettings = FeatureExtraction::createTemporalCovariateSettings(useConditionOcc
   = TRUE, useConditionEraStart = TRUE, useDrugEraStart = TRUE, useProcedureOccurrence =
   TRUE, useMeasurement = TRUE, useObservation = TRUE, temporalStartDays = c(-365, -30,
     0, 1, 31), temporalEndDays = c(-31, -1, 0, 30, 365)),
  minCellCount = 5,
  incremental = FALSE,
  incrementalFolder = exportFolder
)
```

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. The cohortToCreateFile must be .csv file that is expected to be read into a dataframe object identical to requirements for cohortSetReference argument.
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	Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.
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.
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.
databaseDescription	A short description (several sentences) of the database.
runInclusionStatistics	Generate and export statistic on the cohort inclusion rules?
runIncludedSourceConcepts	Generate and export the source concepts included in the cohorts?

`runOrphanConcepts`
: Generate and export potential orphan concepts?

`runTimeDistributions`
: Generate and export cohort time distributions?

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

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

`runCohortOverlap`
: Generate and export the cohort overlap?

`runCohortCharacterization`
: Generate and export the cohort characterization?

`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?

`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, assuming the cohort definitions are stored in that package using the `ROhdsiWebApi::insertCohortDefinitionSetInPackage`, or (2) [WebApi Mode] By using a WebApi interface to retrieve the cohort definitions.

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.

Note: if the parameters for both Package Mode and WebApi Mode are provided, then Package mode is preferred.

The `cohortSetReference` argument must be a data frame with the following columns:

atlasId The cohort ID in ATLAS.

atlasName The full name of the cohort. This will be shown in the Shiny app.

cohortId The cohort ID to use in the package. Usually the same as the cohort ID in ATLAS.

name A short name for the cohort, to use to create file names. do not use special characters.

`runCohortDiagnosticsUsingExternalCounts` *Run cohort diagnostics using external concept counts*

Description

Runs cohort diagnostics on all (or a subset of) the cohorts, but using external concept counts. The external counts must have the following columns:

concept_id The source or target concept ID.

concept_count The number of records having the concept.

concept_subjects The number of unique persons having the concept.

Usage

```
runCohortDiagnosticsUsingExternalCounts(
  packageName = NULL,
  cohortToCreateFile = "settings/CohortsToCreate.csv",
  baseUrl = NULL,
  cohortSetReference = NULL,
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  cohortIds = NULL,
  conceptCountsDatabaseSchema = cdmDatabaseSchema,
  conceptCountsTable = "concept_counts",
  conceptCountsTableIsTemp = FALSE,
  exportFolder,
  databaseId,
  databaseName = databaseId,
  databaseDescription = "",
  runIncludedSourceConcepts = TRUE,
  runOrphanConcepts = TRUE,
  minCellCount = 5
)
```

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. The cohortToCreateFile must be .csv file that is expected to be read into a dataframe object identical to requirements for `cohortSetReference` argument.
`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`
: Should be used in Oracle to specify a schema where the user has write privileges for storing temporary tables.

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

`conceptCountsDatabaseSchema`
: Schema name where your concept counts table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'. Ignored if `conceptCountsTableIsTemp = TRUE`.

`conceptCountsTable`
: Name of the concept counts table. This table can be created using the `createConceptCountsTable`.

`conceptCountsTableIsTemp`
: Is the concept counts table a temp table?

`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.

`databaseDescription`
: A short description (several sentences) of the database.

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

`runOrphanConcepts`
: Generate and export potential orphan concepts?

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

Details

Currently two ways of executing this function are supported, either (1) [Package Mode] embedded in a study package, assuming the cohort definitions are stored in that package using the `ROhdsiWebApi::insertCohortDefinitionSetInPackage`, or (2) [WebApi Mode] By using a WebApi interface to retrieve the cohort definitions.

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.

Note: if the parameters for both Package Mode and WebApi Mode are provided, then Package mode is preferred.

The `cohortSetReference` argument must be a data frame with the following columns:

atlasId The cohort ID in ATLAS.

atlasName The full name of the cohort. This will be shown in the Shiny app.

cohortId The cohort ID to use in the package. Usually the same as the cohort ID in ATLAS.

name A short name for the cohort, to use to create file names. do not use special characters.

Index