

# Package ‘CohortDiagnostics’

February 2, 2022

**Type** Package

**Title** Diagnostics for OHDSI Cohorts

**Version** 3.0.0

**Date** 2021-11-13

**Maintainer** Gowtham Rao <rao@ohdsi.org>

**Description**

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

**Depends** DatabaseConnector (*i*= 4.0.0),

FeatureExtraction (*i*= 3.1.1),

R (*i*= 4.0.0)

**Imports** Andromeda,

checkmate,

clock,

digest,

dplyr (*i*= 1.0.0),

methods,

ParallelLogger (*i*= 2.0.0),

readr (*i*= 2.0.1),

RJSONIO,

rlang,

SqlRender (*i*= 1.7.0),

stringr,

tidyr (*i*= 1.0.0),

CohortGenerator (*i*= 0.2.0)

**Suggests** CirceR,

DT,

Eunomia,

ggiraph,

ggplot2,

htmltools,

knitr,

lubridate,

pool,

plotly,

purrr,

RColorBrewer,

remotes,

rmarkdown,

RSQLite ( $i=2.2.1$ ),  
 scales,  
 shiny,  
 shinydashboard,  
 shinyWidgets,  
 testthat,  
 withr,  
 zip,  
 ROhdsiWebApi ( $i=1.2.0$ )

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

**License** Apache License

**VignetteBuilder** knitr

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

**BugReports** <https://github.com/OHDSI/CohortDiagnostics/issues>

**RoxygenNote** 7.1.2

**Encoding** UTF-8

**Language** en-US

## R topics documented:

checkInputFileEncoding . . . . .	2
createMergedResultsFile . . . . .	3
createResultsDataModel . . . . .	3
executeDiagnostics . . . . .	4
getCohortCounts . . . . .	8
getResultsDataModelSpecifications . . . . .	8
launchCohortExplorer . . . . .	9
launchDiagnosticsExplorer . . . . .	10
uploadResults . . . . .	11

---

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.

---

createMergedResultsFile

*Merge Shiny diagnostics files into sqlite database*

---

**Description**

This function combines diagnostics results from one or more databases into a single file. The result is an sqlite database 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**

```
createMergedResultsFile(
  dataFolder,
  sqliteDbPath = "MergedCohortDiagnosticsData.sqlite",
  overwrite = FALSE
)
```

**Arguments**

dataFolder            folder where the exported zip files for the diagnostics are stored. Use the [executeDiagnostics](#) function to generate these zip files. Zip files containing results from multiple databases may be placed in the same folder.

sqliteDbPath          Output path where sqlite database is placed

overwrite            (Optional) overwrite existing sqlite lite db if it exists.

---

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 <code>connection</code> as created using the <code>connect</code> function in the DatabaseConnector package. Can be left NULL if <code>connectionDetails</code> 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 <code>connectionDetails</code> as created using the <code>createConnectionDetails</code> function in the DatabaseConnector package. Can be left NULL if <code>connection</code> is provided.
schema	The schema on the postgres server where the tables will be created.

## Details

Only PostgreSQL servers are supported.

---

<code>executeDiagnostics</code>	<i>Execute cohort diagnostics</i>
---------------------------------	-----------------------------------

---

## Description

Runs the cohort diagnostics on all (or a subset of) the cohorts instantiated using the CohortGenerator package. Assumes the cohorts have already been instantiated.

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

## Usage

```
executeDiagnostics(
  cohortDefinitionSet,
  exportFolder,
  databaseId,
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
  cohortDatabaseSchema,
  cohortTable = "cohort",
  cohortTableNames = CohortGenerator::getCohortTableNames(cohortTable = cohortTable),
  vocabularyDatabaseSchema = cdmDatabaseSchema,
  cohortIds = NULL,
  inclusionStatisticsFolder = NULL,
  databaseName = databaseId,
  databaseDescription = databaseId,
  cdmVersion = 5,
  runInclusionStatistics = TRUE,
  runIncludedSourceConcepts = TRUE,
  runOrphanConcepts = TRUE,
  runTimeDistributions = TRUE,
  runVisitContext = TRUE,
```

```

runBreakdownIndexEvents = TRUE,
runIncidenceRate = TRUE,
runTimeSeries = FALSE,
runCohortOverlap = TRUE,
runCohortCharacterization = TRUE,
covariateSettings = createDefaultCovariateSettings(),
runTemporalCohortCharacterization = TRUE,
temporalCovariateSettings = createTemporalCovariateSettings(useConditionOccurrence =
  TRUE, useDrugEraStart = TRUE, useProcedureOccurrence = TRUE, useMeasurement = TRUE,
  temporalStartDays = c(-365, -30, 0, 1, 31), temporalEndDays = c(-31, -1, 0, 30, 365)),
minCellCount = 5,
incremental = FALSE,
incrementalFolder = file.path(exportFolder, "incremental")
)

```

## Arguments

<b>cohortDefinitionSet</b>	Data.frame of cohorts must include columns cohortId, cohortName, json, sql
<b>exportFolder</b>	The folder where the output will be exported to. If this folder does not exist it will be created.
<b>databaseId</b>	A short string for identifying the database (e.g. 'Synpuf').
<b>connectionDetails</b>	An object of type <code>connectionDetails</code> as created using the <a href="#">createConnectionDetails</a> function in the DatabaseConnector package. Can be left NULL if connection is provided.
<b>connection</b>	An object of type <code>connection</code> as created using the <a href="#">connect</a> function in the DatabaseConnector package. Can be left NULL if <code>connectionDetails</code> is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
<b>cdmDatabaseSchema</b>	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'.
<b>tempEmulationSchema</b>	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.
<b>cohortDatabaseSchema</b>	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'.
<b>cohortTable</b>	Name of the cohort table.
<b>cohortTableNames</b>	Cohort Table names used by CohortGenerator package
<b>vocabularyDatabaseSchema</b>	Schema name where your OMOP vocabulary data resides. This is commonly the same as <code>cdmDatabaseSchema</code> . Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.

<code>cohortIds</code>	Optionally, provide a subset of cohort IDs to restrict the diagnostics to.
<code>inclusionStatisticsFolder</code>	The folder where the inclusion rule statistics are stored. Can be left NULL if <code>runInclusionStatistics = FALSE</code> .
<code>databaseName</code>	The full name of the database. If NULL, defaults to <code>databaseId</code> .
<code>databaseDescription</code>	A short description (several sentences) of the database. If NULL, defaults to <code>databaseId</code> .
<code>cdmVersion</code>	The version of the OMOP CDM. Default 5. (Note: only 5 is supported.)
<code>runInclusionStatistics</code>	Generate and export statistic on the cohort inclusion rules?
<code>runIncludedSourceConcepts</code>	Generate and export the source concepts included in the cohorts?
<code>runOrphanConcepts</code>	Generate and export potential orphan concepts?
<code>runTimeDistributions</code>	Generate and export cohort time distributions?
<code>runVisitContext</code>	Generate and export index-date visit context?
<code>runBreakdownIndexEvents</code>	Generate and export the breakdown of index events?
<code>runIncidenceRate</code>	Generate and export the cohort incidence rates?
<code>runTimeSeries</code>	Generate and export the cohort prevalence rates?
<code>runCohortOverlap</code>	Generate and export the cohort overlap? Overlaps are checked within <code>cohortIds</code> that have the same phenotype ID sourced from the <code>CohortSetReference</code> or <code>cohortToCreateFile</code> .
<code>runCohortCharacterization</code>	Generate and export the cohort characterization? Only records with values greater than 0.0001 are returned.
<code>covariateSettings</code>	Either an object of type <code>covariateSettings</code> as created using one of the <code>createCovariateSettings</code> function in the <code>FeatureExtraction</code> package, or a list of such objects.
<code>runTemporalCohortCharacterization</code>	Generate and export the temporal cohort characterization? Only records with values greater than 0.001 are returned.
<code>temporalCovariateSettings</code>	Either an object of type <code>covariateSettings</code> as created using one of the <code>createTemporalCovariateSettings</code> function in the <code>FeatureExtraction</code> package, or a list of such objects.
<code>minCellCount</code>	The minimum cell count for fields contains person counts or fractions.
<code>incremental</code>	Create only cohort diagnostics that haven't been created before?
<code>incrementalFolder</code>	If <code>incremental = TRUE</code> , specify a folder where records are kept of which cohort diagnostics has been executed.

## Details

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

**cohortId** The cohort Id is the id used to identify a cohort definition. This is required to be unique. It will be used to create file names. It is recommended to be  $(\text{referentConceptId} * 1000) + \text{a number between 3 to 999}$

**atlasId** Cohort Id in the webApi/atlas instance. It is a required field to run Cohort Diagnostics in WebApi mode. It is discarded in package mode.

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

**logicDescription** A human understandable brief description of the cohort definition. This logic does not have to a fully specified description of the cohort definition, but should provide enough context to help user understand the meaning of the cohort definition

**referentConceptId** A standard omop concept id that serves as the referent phenotype definition for the cohort Id (optional)

## Examples

```
## Not run:
# Load cohorts (assumes that they have already been instantiated)
cohortTableNames <- CohortGenerator::getCohortTableNames(cohortTable = "cohort")
cohorts <- CohortGenerator::getCohortDefinitionSet(packageName = "MyGreatPackage")
connectionDetails <- createConnectionDetails(dbms = "postgresql",
                                             server = "ohdsi.com",
                                             port = 5432,
                                             user = "me",
                                             password = "secure")

executeDiagnostics(cohorts = cohorts,
                   exportFolder = "export",
                   cohortTableNames = cohortTableNames,
                   cohortDatabaseSchema = "results",
                   cdmDatabaseSchema = "cdm",
                   databaseId = "mySpecialCdm",
                   connectionDetails = connectionDetails)

# Use a custom set of cohorts defined in a data.frame
cohorts <- data.frame(
  cohortId = c(100),
  cohortName = c("Cohort Name"),
  logicDescription = c("My Cohort"),
  sql = c(readLines("path_to.sql")),
  json = c(readLines("path_to.json"))
)
executeDiagnostics(cohorts = cohorts,
                   exportFolder = "export",
                   cohortTable = "cohort",
                   cohortDatabaseSchema = "results",
                   cdmDatabaseSchema = "cdm",
                   databaseId = "mySpecialCdm",
                   connectionDetails = connectionDetails)

## End(Not run)
```

---

getCohortCounts	<i>Count the cohort(s)</i>
-----------------	----------------------------

---

### 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 <code>connectionDetails</code> as created using the <a href="#">createConnectionDetails</a> function in the DatabaseConnector package. Can be left NULL if <code>connection</code> is provided.
connection	An object of type <code>connection</code> as created using the <a href="#">connect</a> function in the DatabaseConnector package. Can be left NULL if <code>connectionDetails</code> 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

---

getResultsDataModelSpecifications	<i>Get specifications for Cohort Diagnostics results data model</i>
-----------------------------------	---

---

### Description

Get specifications for Cohort Diagnostics results data model

### Usage

```
getResultsDataModelSpecifications()
```



**Value**

A tibble data frame object with specifications

---

launchCohortExplorer	<i>Launch the CohortExplorer Shiny app</i>
----------------------	--

---

**Description**

Launch the CohortExplorer Shiny app

**Usage**

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

**Arguments**

connectionDetails	An object of type <code>connectionDetails</code> as created using the <a href="#">createConnectionDetails</a> 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(
  sqliteDbPath = "MergedCohortDiagnosticsData.sqlite",
  connectionDetails = NULL,
  resultsDatabaseSchema = NULL,
  vocabularyDatabaseSchema = NULL,
  vocabularyDatabaseSchemas = resultsDatabaseSchema,
  aboutText = NULL,
  runOverNetwork = FALSE,
  port = 80,
  launch.browser = FALSE
)
```

## Arguments

- |                                  |   |
|----------------------------------|---|
| <b>sqliteDbPath</b>              | Path to merged sqlite file. See <a href="#">createMergedResultsFile</a> to create file.   |
| <b>connectionDetails</b>         | An object of type <code>connectionDetails</code> as created using the <a href="#">createConnectionDetails</a> function in the <code>DatabaseConnector</code> package, specifying how to connect to the server where the CohortDiagnostics results have been uploaded using the <a href="#">uploadResults</a> function.  |
| <b>resultsDatabaseSchema</b>     | The schema on the database server where the CohortDiagnostics results have been uploaded.   |
| <b>vocabularyDatabaseSchema</b>  | (Deprecated) Please use <code>vocabularyDatabaseSchemas</code> .  |
| <b>vocabularyDatabaseSchemas</b> | (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 <code>resultsDatabaseSchema</code> . 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. Not supported with an sqlite database. |
| <b>aboutText</b>                 | 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.  |
| <b>runOverNetwork</b>            | (optional) Do you want the app to run over your network?  |
| <b>port</b>                      | (optional) Only used if <code>runOverNetwork = TRUE</code> .  |
| <b>launch.browser</b>            | 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

---

uploadResults	<i>Upload results to the database server.</i>
---------------	---

---

## 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 <code>connectionDetails</code> as created using the <a href="#">createConnectionDetails</a> 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.