Package ‘CdmOnboarding’

January 11, 2025

Title Generate Onboarding Report on an OMOP CDM Instance

Version 3.3.3.900

Maintainer Maxim Moinat <m.moinat@erasmusmc.nl>

Description The goal the CdmOnboarding package is to generate a report to provide insight into the completeness, transparency and quality of the records in an OMOP CDM instance. It connects to the OMOP CDM instance in a database and outputs a Word document.

License Apache License (>= 2)

URL https://github.com/darwin-eu/CdmOnboarding

BugReports https://github.com/darwin-eu/CdmOnboarding/issues

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.0

Depends R (>= 4.0.0)

Imports DatabaseConnector (>= 5.0.2),
ParallelLogger,
SqlRender,
DrugExposureDiagnostics (>= 1.0.6),
CDMConnector (>= 1.0.0),
DBI,
benchmarkme,
prettyunits,
dplyr,
tidyr,
officer,
data.table,
lubridate,
ggplot2,
jsonlite,
stats,
cowplot,
scales,
stringr,
httr,
utils,
tictoc

Suggests RPostgres,
duckdb,
devtools,
testthat (>= 3.0.0),
withr,
Achilles

Remotes OHDSI/Achilles

Config/testthat/edition 3

Contents

`.exportDedResults` *Export DrugExposureDiagnostics results to csv file*

Description

Export DrugExposureDiagnostics results to csv file

Usage

```
.exportDedResults(results, outputFolder = getwd())
```

Arguments

`results` results object from `cdmOnboarding`

`outputFolder` folder to store the results

`.getCdmConnection` *Convenience function to get a CDMConnector connection from connectionDetails*

Description

Convenience function to get a CDMConnector connection from connectionDetails

Usage

```
.getCdmConnection(connectionDetails)
```

Arguments

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

Value

CDMConnector connection

`.getCdmSource` *Get CDM source table*

Description

Get CDM source table

Usage

```
.getCdmSource(connectionDetails, cdmDatabaseSchema, outputFolder)
```

Arguments

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

`cdmDatabaseSchema`
Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'.

`outputFolder` Path to store logs and SQL files

Value

```
A data frame with the CDM source table
```

.processDqdResults *Process dqd file*

Description

Process dqd file

Usage

```
.processDqdResults(dqdJsonPath)
```

Arguments

dqdJsonPath	Path to the DQD JSON file

Value

list with version, overview, startTimestamp, executionTime

.runBenchmarkCdmConnector *Run Benchmark CDMConnector*

Description

Run Benchmark CDMConnector

Usage

```
.runBenchmarkCdmConnector(
  connectionDetails,
  cdmDatabaseSchema,
  scratchDatabaseSchema
)
```

Arguments

connectionDetails	An R object of type connectionDetails created using the function createConnectionDetails in the DatabaseConnector package.
cdmDatabaseSchema	Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'.
scratchDatabaseSchema	Fully qualified name of database schema where temporary tables can be written.

Value

list of DED diagnostics_summary and duration

`.runCohortBenchmark` *Generate predefined set of cohorts*

Description

Generate predefined set of cohorts

Usage

```
.runCohortBenchmark(
  connectionDetails,
  cdmDatabaseSchema,
  scratchDatabaseSchema
)
```

Arguments

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

`cdmDatabaseSchema`
Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'.

`scratchDatabaseSchema`
Fully qualified name of database schema where temporary tables can be written.

Value

list of DED diagnostics_summary and duration

`.runDedChecks` *Run DrugExposureDiagnostics for a set of default ingredient concepts*

Description

Run DrugExposureDiagnostics for a set of default ingredient concepts

Usage

```
.runDedChecks(connectionDetails, cdmDatabaseSchema, scratchDatabaseSchema)
```

Arguments

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

`cdmDatabaseSchema`
: Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'.

`scratchDatabaseSchema`
: Fully qualified name of database schema where temporary tables can be written.

Value

list of DED diagnostics_summary and duration

`bundleResults` *Bundles the results in a zip file*

Description

`bundleResults` creates a zip file with results in the outputFolder

Usage

```
bundleResults(outputFolder, databaseId)
```

Arguments

`outputFolder`	Folder to store the results
`databaseId`	ID of your database, this will be used as subfolder for the results.

`cdmOnboarding` *The main CDM Onboarding method (for v5.x)*

Description

`cdmOnboarding` runs the CDM Onboarding procedure. Executing the checks and outputing a results document

Usage

```
cdmOnboarding(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  scratchDatabaseSchema = resultsDatabaseSchema,
  oracleTempSchema = resultsDatabaseSchema,
  databaseId,
  databaseName,
  databaseDescription,
  authors = "",
  runVocabularyChecks = TRUE,
  runDataTablesChecks = TRUE,
  runPerformanceChecks = TRUE,
  runWebAPIChecks = TRUE,
  runDedChecks = TRUE,
  runCohortBenchmarkChecks = TRUE,
  smallCellCount = 5,
  baseUrl = NULL,
  sqlOnly = FALSE,
  outputFolder = "output",
  verboseMode = TRUE,
  dqdJsonPath = NULL,
  optimize = FALSE,
  dedIngredientIds = NULL
)
```

Arguments

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

`cdmDatabaseSchema`
Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'.

`resultsDatabaseSchema`
Fully qualified name of database schema that we can write final results to. Default is cdmDatabaseSchema. The Achilles results are read from this table. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_results.dbo'.

`scratchDatabaseSchema`
Fully qualified name of database schema that we can write temporary tables to. Default is resultsDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_scratch.dbo'.

`oracleTempSchema`
For Oracle only: the name of the database schema where you want all temporary tables to be managed. Requires create/insert permissions to this database.

`databaseId` ID of your database, this will be used as subfolder for the results and naming of the report

`databaseName` String name of the database name. If blank, CDM_SOURCE table will be queried to try to obtain this.

`databaseDescription`
Provide a short description of the database. If blank, CDM_SOURCE table will be queried to try to obtain this.

`authors` List of author names to be added in the document

`runVocabularyChecks`
Boolean to determine if vocabulary checks need to be run. Default = TRUE

`runDataTablesChecks`
Boolean to determine if table checks need to be run. Default = TRUE

`runPerformanceChecks`
Boolean to determine if performance checks need to be run. Default = TRUE

`runWebAPIChecks`
Boolean to determine if WebAPI checks need to be run. Default = TRUE

`runDedChecks` Boolean to determine if DrugExposureDiagnostics checks need to be run. Default = TRUE

`runCohortBenchmarkChecks`
Boolean to determine if CohortBenchMark checks need to be run. Default = TRUE

`smallCellCount` To avoid patient identifiability, source values with small counts (<= smallCellCount) are deleted. Set to NULL if you don't want any deletions. (default 5)

`baseUrl` WebAPI url, example: http://server.org:80/WebAPI

`sqlOnly` If TRUE, queries will be written to file instead of executed. Not supported for DED Checks.

`outputFolder` Path to store logs and SQL files

`verboseMode` Boolean to determine if the console will show all execution steps. Default = TRUE

`dqdJsonPath` Path to the json of the DQD

`optimize` Boolean to determine if heuristics will be used to speed up execution. Currently only implemented for postgresql databases. Default = FALSE

`dedIngredientIds`
DEPRECATED, default ingredients are always used (`getDedIngredients()`).

Details

`cdmOnboarding` runs the CDM Onboarding procedure. Executing the checks and outputing a results document

Value

```
An object of type \code{achillesResults} containing details for connecting to th
```

`compat` *Compatibility of CdmOnboarding results*

Description

`compat` provides compatibility by converting results from previous versions of CdmOnboarding to the latest version.

Usage

```
compat(r, target_version = package_version("3.0"))
```

Arguments

`r` A list of results from CdmOnboarding.

`target_version` The version to convert the results to, only v3 supported.

Value

A list of results from CdmOnboarding, forwards compatible with v3.

dataTablesChecks *The Data Tables checks (for v5.x)*

Description

dataTablesChecks runs a list of checks on the clinical data tables as part of the CDM Onboarding procedure

Usage

```
dataTablesChecks(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  cdmVersion,
  sqlOnly = FALSE,
  outputFolder = "output",
  optimize = FALSE
)
```

Arguments

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

cdmDatabaseSchema
: Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'.

resultsDatabaseSchema
: Fully qualified name of database schema that we can write final results to. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_results.dbo'.

cdmVersion
: Define the OMOP CDM version used: currently supports v5 and above. Use major release number or minor number only (e.g. 5, 5.3)

sqlOnly
: Boolean to determine if Achilles should be fully executed. TRUE = just generate SQL files, don't actually run, FALSE = run Achilles

outputFolder
: Path to store logs and SQL files

optimize
: Boolean to determine if heuristics will be used to speed up execution. Currently only implemented for postgresql databases. Default = FALSE

Details

dataTablesChecks runs a list of checks on the clinical data tables as part of the CDM Onboarding procedure

Value

```
An object of type \code{achillesResults} containing details for connecting to the
```

executeQuery — *Execute sql file on the given connectionDetails*

Description

Execute sql file on the given connectionDetails

Usage

```
executeQuery(
  outputFolder,
  sqlFileName,
  successMessage = NULL,
  connectionDetails = NULL,
  sqlOnly = FALSE,
  activeConnection = NULL,
  useExecuteSql = FALSE,
  ...
)
```

Arguments

`outputFolder` Path to store logs and SQL files

`sqlFileName` Name of the SQL file to execute

`successMessage` Message to log when the query is successful

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

`sqlOnly` Boolean indicating if only the SQL should be written to file

`activeConnection`
An active connection object to use

`useExecuteSql` Boolean indicating if the query should be executed using `executeSql` instead of `querySql`

`...` Additional parameters to pass to SqlRender::loadRenderTranslateSql

Value

result of the query

exportDedResults *Export DrugExposureDiagnostics results to csv file in same folder as input file path*

Description

Export DrugExposureDiagnostics results to csv file in same folder as input file path

Usage

```
exportDedResults(path)
```

Arguments

`path`	path to the CdmOnboarding .rds results file, output is written to the same folder

generateAppendixSection *Generates the Appendix section for the Results Document*

Description

Generates the Appendix section for the Results Document

Usage

```
generateAppendixSection(doc, df, optimized)
```

Arguments

`doc`	officer document object to add the section to
`df`	Results object from `cdmOnboarding` vocabularyResults
`optimized`	boolean indicating if the optimized queries were used

generateCohortBenchmarkSection *Generates the Cohort Benchmark section for the Results Document*

Description

Generates the Cohort Benchmark section for the Results Document

Usage

```
generateCohortBenchmarkSection(doc, df)
```

Arguments

`doc`	officer document object to add the section to
`df`	Results object from `cdmOnboarding`

generateDataTablesSection *Generates the Data Tables section for the Results Document*

Description

Generates the Data Tables section for the Results Document

Usage

```
generateDataTablesSection(doc, df, cdmSource, optimized)
```

Arguments

doc	officer document object to add the section to
df	Results object from `cdmOnboarding` dataTablesResults
cdmSource	CDM source object
optimized	boolean indicating if the optimized queries were used

generateDedSection *Generates the Drug Exposure Diagnostics section for the Results Document*

Description

Generates the Drug Exposure Diagnostics section for the Results Document

Usage

```
generateDedSection(doc, df)
```

Arguments

doc	officer document object to add the section to
df	Results object from `cdmOnboarding`

`generateDqdSection` *Generates the DQD section for the Results Document*

Description

Generates the DQD section for the Results Document

Usage

```
generateDqdSection(doc, df)
```

Arguments

`doc`	officer document object to add the section to
`df`	Results object from `cdmOnboarding`

`generateExecutionDetails` *Generates the Execution Details section for the Results Document*

Description

Generates the Execution Details section for the Results Document

Usage

```
generateExecutionDetails(doc, df)
```

Arguments

`doc`	officer document object to add the section to
`df`	Results object from `cdmOnboarding`

`generatePerformanceSection` *Generates the Performance section in the Results Document*

Description

Generates the Performance section in the Results Document

Usage

```
generatePerformanceSection(doc, results)
```

Arguments

`doc`	officer document object to add the section to
`results`	Results object from `cdmOnboarding`

generateResultsDocument — *Generates the Results Document*

Description

`generateResultsDocument` creates a word document with results based on a template

Usage

```
generateResultsDocument(results, outputFolder, authors)
```

Arguments

`results`	Results object from `cdmOnboarding`
`outputFolder`	Folder to store the results
`authors`	List of author names to be added in the document

generateVocabularySection — *Generates the Vocabulary Section in the Results Document*

Description

Generates the Vocabulary Section in the Results Document

Usage

```
generateVocabularySection(doc, df, smallCellCount)
```

Arguments

`doc`	officer document object to add the section to
`df`	Results object from `cdmOnboarding`
`smallCellCount`	number of cells to display in the unmapped/mapped tables

getDARWINpackages — *Hard coded list of DARWIN EU® packages that CdmOnboarding checks against.*

Description

Hard coded list of DARWIN EU® packages that CdmOnboarding checks against.

Usage

```
getDARWINpackages()
```

Value

character vector with DARWIN EU® package names

`getDedIngredients` *Returns data frame with concept_id and concept_name of drug ingredients used for the DrugExposureDiagnostics check*

Description

Returns data frame with concept_id and concept_name of drug ingredients used for the DrugExposureDiagnostics check

Usage

```
getDedIngredients()
```

`getHADESpackages` *Hard coded list of HADES packages that CdmOnboarding checks against. Does NOT update automatically when new HADES packages are released.*

Description

Hard coded list of HADES packages that CdmOnboarding checks against. Does NOT update automatically when new HADES packages are released.

Usage

```
getHADESpackages()
```

Value

character vector with HADES package names

`getMappingLevel` *Get drug class levels from DrugExposureDiagnostics results*

Description

Get drug class levels from DrugExposureDiagnostics results

Usage

```
getMappingLevel(dedResults)
```

Arguments

`dedResults`	DrugExposureDiagnostics results object

Value

data frame with for each ingredient and concept_class_id the number of concepts and records

performanceChecks *The performane checks (for v5.x)*

Description

`PerformanceChecks` runs a list of performance checks as part of the CDM Onboarding procedure

Usage

```
performanceChecks(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  scratchDatabaseSchema,
  cdmVersion = "5.4",
  sqlOnly = FALSE,
  outputFolder = "output"
)
```

Arguments

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

`cdmDatabaseSchema`
Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'.

`resultsDatabaseSchema`
Fully qualified name of database schema that we can write final results to. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_results.dbo'.

`scratchDatabaseSchema`
Fully qualified name of database schema where temporary tables can be written.

`cdmVersion` Version of the CDM to check against. Default is "5.4".

`sqlOnly` Boolean to determine if Achilles should be fully executed. TRUE = just generate SQL files, don't actually run, FALSE = run Achilles

`outputFolder` Path to store logs and SQL files

Details

`PerformanceChecks` runs a list of performance checks as part of the CDM Onboarding procedure

Value

```
An object of type \code{achillesResults} containing details for connecting to the
```

`vocabularyChecks` *The vocabulary checks (for v5.x)*

Description

`vocabularyChecks` runs a list of checks on the vocabulary as part of the CDM Onboarding procedure

Usage

```
vocabularyChecks(
  connectionDetails,
  cdmDatabaseSchema,
  cdmVersion,
  smallCellCount = 5,
  sqlOnly = FALSE,
  outputFolder = "output",
  optimize = FALSE
)
```

Arguments

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

`cdmDatabaseSchema`
Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'.

`cdmVersion` Define the OMOP CDM version used: currently supports v5 and above. Use major release number or minor number only (e.g. 5, 5.3)

`smallCellCount` To avoid patient identifiability, cells with small counts (<= smallCellCount) are deleted. Set to NULL if you don't want any deletions.

`sqlOnly` Boolean to determine if Achilles should be fully executed. TRUE = just generate SQL files, don't actually run, FALSE = run Achilles

`outputFolder` Path to store logs and SQL files

`optimize` Boolean to determine if heuristics will be used to speed up execution. Currently only implemented for postgresql databases. Default = FALSE

Details

`vocabularyChecks` runs a list of checks on the vocabulary as part of the CDM Onboarding procedure

Value

```
An object of type \code{achillesResults} containing details for connecting to the
```

Index