Package 'DrugExposureDiagnostics'

August 16, 2023

Title Diagnostics for OMOP Common Data Model Drug Records

Version 0.4.6

Description Ingredient specific diagnostics for drug exposure records in the Observational Medical Outcomes Partnership (OMOP) common data model.

License Apache License (>= 2)

Encoding UTF-8

RoxygenNote 7.2.3

Depends R (>= 4.0)

Imports CDMConnector (>= 1.0.0), dplyr (>= 1.0.0), magrittr (>= 2.0.0), rlang (>= 1.0.0), tidyr (>= 1.2.0), tidyselect (>= 1.2.0), checkmate (>= 2.0.0), glue (>= 1.5.0)

Suggests testthat (>= 3.0.0), duckdb, odbc, DBI, knitr, rmarkdown, zip, lubridate, tibble, DT, graphics, SqlRender, PaRe, magick, DiagrammeRsvg, ggplot2, cowplot, plotly

Config/testthat/edition 3

VignetteBuilder knitr

Language en-US

NeedsCompilation no

Author Ger Inberg [aut, cre] (<https://orcid.org/0000-0001-8993-8748>), Edward Burn [aut] (<https://orcid.org/0000-0002-9286-1128>), Yuchen Guo [ctb] (<https://orcid.org/0000-0002-0847-4855>), Marti Catala [ctb] (<https://orcid.org/0000-0003-3308-9905>), Mike Du [ctb] (<https://orcid.org/0000-0002-9517-8834>), Theresa Burkard [ctb] (<https://orcid.org/0000-0003-1313-4473>), Xintong Li [ctb] (<https://orcid.org/0000-0002-6872-5804>), Ross Williams [ctb] (<https://orcid.org/0000-0001-7723-417X>), Erasmus MC [cph]

Maintainer Ger Inberg <g.inberg@erasmusmc.nl>

Repository CRAN

Date/Publication 2023-08-16 14:34:35 UTC

R topics documented:

`checkDaysSupply` *Check if Days_supply is the same as datediff(drug_exp_start_date,drug_exp_end_date)*

Description

Check if Days_supply is the same as datediff(drug_exp_start_date,drug_exp_end_date)

Usage

```
checkDaysSupply(cdm, drugRecordsTable = "drug_exposure", byConcept = TRUE)
```

Arguments

cdm	CDMConnector reference object
drugRecordsTable	drug exposure table
byConcept	whether to get result by concept

Value

a table with the stats of days supply compared to start and end date

---

checkDbType *Check the database type.*

Description

Check the database type.

Usage

```
checkDbType(cdm, type = "cdm_reference", messageStore)
```

Arguments

cdm	CDMConnector reference object
type	type of the database, default cdm_reference
messageStore	checkmate collection

---

checkDrugDose *Get a summary of the daily drug dose*

Description

Get a summary of the daily drug dose

Usage

```
checkDrugDose(
  cdm,
  drugRecordsTable = "ingredient_drug_records",
  drugStrengthTable = "drug_strength",
  byConcept = TRUE
)
```

Arguments

cdm	CDMConnector reference object
drugRecordsTable	drug exposure table
drugStrengthTable	drug strength table
byConcept	whether to get result by concept

Value

a table with the stats about the daily dose

---

checkDrugSig *Check the drug sig field; this is the verbatim instruction for the drug as written by the provider.*

Description

Check the drug sig field; this is the verbatim instruction for the drug as written by the provider.

Usage

```
checkDrugSig(cdm, drugRecordsTable = "drug_exposure", byConcept = TRUE)
```

Arguments

cdm	CDMConnector reference object
drugRecordsTable	drug exposure table
byConcept	whether to get result by drug concept

Value

a table with a summary of the sig values

checkIsIngredient *Check is an ingredient*

Description

Check is an ingredient

Usage

```
checkIsIngredient(cdm, conceptId, messageStore)
```

Arguments

cdm	CDMConnector reference object
conceptId	ingredient concept id to check
messageStore	checkmate collection

checkLogical *Check if given object is a boolean.*

Description

Check if given object is a boolean.

Usage

```
checkLogical(input, messageStore, null.ok = TRUE)
```

Arguments

input	the input
messageStore	checkmate collection
null.ok	if value null is allowed

checkTableExists *Check if given table exists in cdm.*

Description

Check if given table exists in cdm.

Usage

```
checkTableExists(cdm, tableName, messageStore)
```

Arguments

cdm	CDMConnector reference object
tableName	checkmate collection
messageStore	the message store

checkVerbatimEndDate *Check the verbatim_end_date field*

Description

Check the verbatim_end_date field

Usage

```
checkVerbatimEndDate(cdm, drugRecordsTable = "drug_exposure", byConcept = TRUE)
```

Arguments

cdm	CDMConnector reference object
drugRecordsTable	drug exposure table
byConcept	whether to get result by concept

Value

a table with the stats about the verbatim_end_date

`computeDBQuery` *Store the given input in a remote database table. It will be stored either in a permanent table or a temporary table depending on tablePrefix.*

Description

Store the given input in a remote database table. It will be stored either in a permanent table or a temporary table depending on tablePrefix.

Usage

```
computeDBQuery(table, tablePrefix, tableName, cdm, overwrite = TRUE)
```

Arguments

`table`	the input table
`tablePrefix`	The stem for the permanent tables that will be created when running the diagnostics. Permanent tables will be created using this prefix, and any existing tables that start with this will be at risk of being dropped or overwritten. If NULL, temporary tables will be used throughout.
`tableName`	the input table
`cdm`	cdm reference object
`overwrite`	if the table should be overwritten (default TRUE).

Value

reference to the table

`createHistogram` *create a histogram for one of days_supply, duration, quantity*

Description

create a histogram for one of days_supply, duration, quantity

Usage

```
createHistogram(cdm, drugRecordsTable = "drug_exposure", type)
```

Arguments

`cdm`	CDMConnector reference object
`drugRecordsTable`	drug exposure table
`type`	specify whether to plot days_supply, duration or quantity

Value

object containing a histogram

`dataFrame2Hist` *Load an object of class histogram from a data.frame*

Description

Load an object of class histogram from a data.frame

Usage

```
dataFrame2Hist(df)
```

Arguments

`df`	the dataframe

Value

an object of class histogram

`executeChecks` *Execute all checks on Drug Exposure.*

Description

Execute all checks on Drug Exposure.

Usage

```
executeChecks(
  cdm,
  ingredients = c(1125315),
  subsetToConceptId = NULL,
  checks = c("missing", "exposureDuration", "type", "route", "sourceConcept",
    "daysSupply", "verbatimEndDate", "dose", "sig", "quantity", "histogram"),
  minCellCount = 5,
  sample = 1e+06,
  tablePrefix = NULL,
  earliestStartDate = "2010-01-01",
  verbose = FALSE
)
```

Arguments

cdm	CDMConnector reference object
ingredients	vector of ingredients, by default: acetaminophen
subsetToConceptId	vector of concept IDs of the ingredients to subset down to. If NULL, all concept IDs for an ingredient will be considered.
checks	the checks to be executed, by default everything
minCellCount	minimum number of events to report- results lower than this will be obscured. If NULL all results will be reported.
sample	the number of samples, default 1 million
tablePrefix	The stem for the permanent tables that will be created when running the diagnostics. Permanent tables will be created using this prefix, and any existing tables that start with this will be at risk of being dropped or overwritten. If NULL, temporary tables will be used throughout.
earliestStartDate	the earliest date from which a record can be included
verbose	verbose, default FALSE

Value

named list with results

Examples

```
## Not run:
db <- DBI::dbConnect(" Your database connection here ")
cdm <- CDMConnector::cdm_from_con(
  con = db,
  cdm_schema = "cdm schema name"
)
result <- executeChecks(
  cdm = cdm,
  ingredients = c(1125315))

## End(Not run)
```

---

executeChecksSingleIngredient

Execute all checks on Drug Exposure for a single ingredient.

Description

Execute all checks on Drug Exposure for a single ingredient.

Usage

```
executeChecksSingleIngredient(
  cdm,
  ingredient = 1125315,
  subsetToConceptId = NULL,
  checks = c("missing", "exposureDuration", "type", "route", "sourceConcept",
    "daysSupply", "verbatimEndDate", "dose", "sig", "quantity", "histogram"),
  minCellCount = 5,
  sample = 1e+06,
  tablePrefix = NULL,
  earliestStartDate = "2010-01-01",
  verbose = FALSE
)
```

Arguments

cdm	CDMConnector reference object
ingredient	ingredient, by default: acetaminophen
subsetToConceptId	vector of concept IDs of the ingredients to subset down to. If NULL, all concept IDs for an ingredient will be considered.
checks	the checks to be executed, by default everything
minCellCount	minimum number of events to report- results lower than this will be obscured. If NULL all results will be reported.
sample	the number of samples, default 1 million
tablePrefix	The stem for the permanent tables that will be created when running the diagnostics. Permanent tables will be created using this prefix, and any existing tables that start with this will be at risk of being dropped or overwritten. If NULL, temporary tables will be used throughout.
earliestStartDate	the earliest date from which a record can be included
verbose	verbose, default FALSE

Value

named list with results

---

getDrugMissings *Check missings in drug exposure records*

Description

Check missings in drug exposure records

Usage

```
getDrugMissings(cdm, drugRecordsTable = "drug_exposure", byConcept = TRUE)
```

Arguments

cdm	CDMConnector reference object
drugRecordsTable	drugRecordsTable
byConcept	byConcept

Value

a table with a summary of missing records

---

getDrugRecords *Drug exposure records for ingredients of interest*

Description

Drug exposure records for ingredients of interest

Usage

```
getDrugRecords(
  cdm,
  ingredient,
  includedConceptsTable,
  drugRecordsTable = "drug_exposure",
  tablePrefix = NULL,
  verbose = FALSE
)
```

Arguments

cdm	CDMConnector reference object
ingredient	Concept ID for ingredient of interest
includedConceptsTable	includedConceptsTable
drugRecordsTable	drugRecordsTable, default "drug_exposure"
tablePrefix	The stem for the permanent tables that will be created when running the diagnostics. Permanent tables will be created using this prefix, and any existing tables that start with this will be at risk of being dropped or overwritten. If NULL, temporary tables will be used throughout.
verbose	verbose

Value

a table containing drug exposure records

---

getDrugRoutes *Get drug exposure route types*

Description

Get drug exposure route types

Usage

```
getDrugRoutes(cdm, drugRecordsTable = "drug_exposure", byConcept = TRUE)
```

Arguments

cdm	CDMConnector reference object
drugRecordsTable	drugRecordsTable
byConcept	byConcept

Value

a table with the drug exposure route types

---

getDrugSourceConcepts *Check drug exposure source types*

Description

Check drug exposure source types

Usage

```
getDrugSourceConcepts(
  cdm,
  drugRecordsTable = "drug_exposure",
  byConcept = TRUE
)
```

Arguments

cdm	CDMConnector reference object
drugRecordsTable	drugRecordsTable
byConcept	byConcept

Value

a table with the drug source concepts

`getDrugStrength` *Drug strength records for ingredients of interest*

Description

Drug strength records for ingredients of interest

Usage

```
getDrugStrength(
  cdm,
  ingredient,
  includedConceptsTable,
  drugStrengthTable = "drug_strength",
  tablePrefix = NULL,
  verbose = FALSE
)
```

Arguments

`cdm`	CDMConnector reference object
`ingredient`	ingredient concept ID for ingredient of interest
`includedConceptsTable`	table name for the concept ids, names and units
`drugStrengthTable`	table name for drug strength, default "drug_strength"
`tablePrefix`	The stem for the permanent tables that will be created when running the diagnostics. Permanent tables will be created using this prefix, and any existing tables that start with this will be at risk of being dropped or overwritten. If NULL, temporary tables will be used throughout.
`verbose`	verbose

Value

a table containing drug strength records

getDrugTypes *Get drug exposure record types*

Description

Get drug exposure record types

Usage

```
getDrugTypes(cdm, drugRecordsTable = "drug_exposure", byConcept = TRUE)
```

Arguments

cdm	CDMConnector reference object
drugRecordsTable	drugRecordsTable
byConcept	byConcept

Value

a table with the drug exposure record types

getDuration *Compute the difference in days between 2 variables in a database table.*

Description

Compute the difference in days between 2 variables in a database table.

Usage

```
getDuration(
  cdm,
  tableName = "drug_exposure",
  startDateCol = "drug_exposure_start_date",
  endDateCol = "drug_exposure_end_date",
  colName = "duration"
)
```

Arguments

cdm	CDMConnector reference object
tableName	the table name
startDateCol	the start date column name
endDateCol	the end date column name
colName	the result column name

Value

the table with as new column the duration

`getEunomiaCdm` *Get an eunomia CDM reference for given ingredient*

Description

Get an eunomia CDM reference for given ingredient

Usage

```
getEunomiaCdm(ingredientId = 1125315)
```

Arguments

`ingredientId`	The ingredient concept id

Value

A list of dplyr database table references pointing to CDM tables

Examples

```
## Not run:
library(CDMConnector)
getEunomiaCdm(1125315)

## End(Not run)
```

`getIngredientOverview` *Get a detailed ingredient overview. The record count and patient count will be returned for an unique combination of data elements.*

Description

Get a detailed ingredient overview. The record count and patient count will be returned for an unique combination of data elements.

Usage

```
getIngredientOverview(
  cdm,
  drugRecordsTable = "drug_exposure",
  drugStrengthTable = "drug_strength"
)
```

Arguments

`cdm`	CDMConnector reference object
`drugRecordsTable`	drug exposure table
`drugStrengthTable`	drug strength table

Value

a table with the stats

`getIngredientPresence` *Get a presence overview for the ingredient. The record count and patient count will be returned for a bit set.*

Description

Get a presence overview for the ingredient. The record count and patient count will be returned for a bit set.

Usage

```
getIngredientPresence(
  cdm,
  drugRecordsTable = "drug_exposure",
  drugStrengthTable = "drug_strength"
)
```

Arguments

`cdm`	CDMConnector reference object
`drugRecordsTable`	drug exposure table
`drugStrengthTable`	drug strength table

Value

a table with the bit set

hist2DataFrame *Save an object of class histogram to a data.frame*

Description

Save an object of class histogram to a data.frame

Usage

```
hist2DataFrame(h)
```

Arguments

h	a histogram

Value

a dataframe with the converted values of the histogram

ingredientDescendantsInDb *Get the descendants for the given ingredients*

Description

Get the descendants for the given ingredients

Usage

```
ingredientDescendantsInDb(
  cdm,
  ingredient,
  drugRecordsTable = "drug_exposure",
  tablePrefix = NULL,
  verbose = FALSE
)
```

Arguments

cdm	CDMConnector reference object
ingredient	ingredient concept id for ingredient of interest
drugRecordsTable	table name of the drug exposure records

tablePrefix	The stem for the permanent tables that will be created when running the diagnostics. Permanent tables will be created using this prefix, and any existing tables that start with this will be at risk of being dropped or overwritten. If NULL, temporary tables will be used throughout.
verbose	if verbose set to TRUE, the function will output extra messages

Value

temp table with concepts used

`mockDrugExposure` *Mock Drug exposure tables for ingredients of interest*

Description

Mock Drug exposure tables for ingredients of interest

Usage

```
mockDrugExposure(
  drug_exposure = NULL,
  concept_ancestor = NULL,
  concept_relationship = NULL,
  concept = NULL,
  drug_strength = NULL,
  drug_exposure_size = 100,
  patient_size = 50,
  amount_val = c(1, 2, 3),
  den_val = c(1, 10, 100),
  unit = c("", "actuat", "mg", "mL", "mL", "h"),
  num_val = c(1, 2, 3),
  seed = 1
)
```

Arguments

drug_exposure	drug exposure table
concept_ancestor	concept_ancestor table
concept_relationship	concept_relationship table
concept	concept table
drug_strength	drug strength table
drug_exposure_size	the sample size of the drug exposure table
patient_size	the number of unique patients in the drug exposure table

amount_val	vector of possible numeric amount value for the drug in the drug strength table
den_val	vector of possible numeric denominator value for the drug in drug strength table
unit	vector of possible unit type drug strength table please select from "", "actuat", "mg", "mL", "mL", "h".
num_val	vector of possible numeric numerator denominator value drug strength table
seed	seed to make results reproducible

Value

CDMConnector CDM reference object to duckdb database with mock data include concept_ancestor, concept, drug_strength, drug_exposure tables

---

`obscureCounts` *Obscure the small number of counts*

Description

Obscure the small number of counts

Usage

```
obscureCounts(table, tableName, minCellCount = 5, substitute = NA)
```

Arguments

table	the table as a tibble
tableName	the table name
minCellCount	the minimum number of counts that will be displayed. If NULL all results will be reported.
substitute	the substitute value if values will be obscured

Value

the input table with results obscured if minCellCount applies

`printDurationAndMessage` *Print duration from start to now and print it as well as new status message*

Description

Print duration from start to now and print it as well as new status message

Usage

```
printDurationAndMessage(message, start)
```

Arguments

`message`	the message
`start`	the start time

Value

the current time

`summariseChecks` *Create a summary about the diagnostics results*

Description

Create a summary about the diagnostics results

Usage

```
summariseChecks(resultList)
```

Arguments

`resultList`	a list with the diagnostics results

Value

a table containing the diagnostics summary

summariseDaysSupply *Create a summary of the days_supply field*

Description

Create a summary of the days_supply field

Usage

```
summariseDaysSupply(cdm, drugRecordsTable = "drug_exposure")
```

Arguments

cdm	CDMConnector reference object
drugRecordsTable	table name containing the drug exposure records

Value

a table with the days supply summary

summariseDrugExposureDuration *Summarise drug exposure record durations*

Description

Summarise drug exposure record durations

Usage

```
summariseDrugExposureDuration(
  cdm,
  drugRecordsTable = "drug_exposure",
  byConcept = TRUE
)
```

Arguments

cdm	CDMConnector reference object
drugRecordsTable	drugRecordsTable
byConcept	byConcept

Value

a table with the drug exposure record durations

summariseQuantity *Summarise the quantity column of the drug_exposure table*

Description

Summarise the quantity column of the drug_exposure table

Usage

```
summariseQuantity(cdm, drugRecordsTable = "drug_exposure", byConcept = TRUE)
```

Arguments

cdm	CDMConnector reference object
drugRecordsTable	drug exposure table
byConcept	whether to get result by concept

Value

a table with the summarized quantity result

writeResultToDisk *Write diagnostics results to a zip file on disk in given output folder.*

Description

Write diagnostics results to a zip file on disk in given output folder.

Usage

```
writeResultToDisk(resultList, databaseId, outputFolder, filename = NULL)
```

Arguments

resultList	named list with results
databaseId	database identifier
outputFolder	folder to write to
filename	output filename, if NULL it will be equal to databaseId

Value

No return value, called for side effects

Examples

```
## Not run:
resultList <- list("mtcars" = mtcars)
result <- writeResultToDisk(
  resultList = resultList,
  databaseId = "mtcars",
  outputFolder = here::here())

## End(Not run)
```

Index