# Package ‘PhenotypeR’

December 6, 2024

**Type** Package

**Title** Assess Study Cohorts Using a Common Data Model

**Version** 0.1.0

**Description** Phenotype study cohorts in data mapped to the
Observational Medical Outcomes Partnership Common Data Model. Diagnostics
are run at the database, code list, cohort, and population level to assess
whether study cohorts are ready for research.

**License** Apache License (>= 2)

**Encoding** UTF-8

**Depends** R (>= 4.0)

**Suggests** CDMConnector (>= 1.6.1), duckdb, DBI, gt, omock, testthat (>=
3.0.0), knitr, visOmopResults (>= 0.5.0), glue, RPostgres,
PatientProfiles (>= 1.2.2), ggplot2, ggpubr, stringr, shiny,
DiagrammeR, sortable, shinycssloaders, here, DT, bslib,
shinyWidgets, plotly, tidyr

**Config/testthat/edition** 3

**RoxygenNote** 7.3.2

**Imports** CodelistGenerator (>= 3.3.1), CohortCharacteristics (>=
0.4.0), CohortConstructor (>= 0.3.2), cli, dplyr,
IncidencePrevalence (>= 0.9.0), omopgenerics (>= 0.4.0),
OmopSketch (>= 0.1.2), magrittr, purrr, rlang, vctrs, usethis

**URL** `https://ohdsi.github.io/PhenotypeR/`

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Edward Burn [aut, cre] (<`https://orcid.org/0000-0002-9286-1128`>),
Marti Catala [aut] (<`https://orcid.org/0000-0003-3308-9905`>),
Xihang Chen [aut] (<`https://orcid.org/0009-0001-8112-8959`>),
Marta Alcalde-Herraiz [aut] (<`https://orcid.org/0009-0002-4405-1814`>),
Albert Prats-Uribe [aut] (<`https://orcid.org/0000-0003-1202-9153`>)

**Maintainer** Edward Burn <`edward.burn@ndorms.ox.ac.uk`>

**Repository** CRAN

**Date/Publication** 2024-12-06 09:10:05 UTC

# Contents

---

## `addCodelistAttribute` *Adds the cohort_codelist attribute to a cohort*

---

### Description

‘addCodelistAttribute()‘ allows the users to add a codelist to a cohort in OMOP CDM.

This is particularly important for the use of ‘codelistDiagnostics()‘, as the underlying assumption is that the cohort that is fed into ‘codelistDiagnostics()‘ has a cohort_codelist attribute attached to it.

### Usage

```
addCodelistAttribute(cohort, codelist, cohortName = names(codelist))
```

### Arguments

| | |
|---|---|
| `cohort` | Cohort table in a cdm reference |
| `codelist` | Named list of concepts |
| `cohortName` | For each element of the codelist, the name of the cohort in ‘cohort‘ to which the codelist refers |

### Value

A cohort

### Examples

```
library(IncidencePrevalence)
cdm <- mockIncidencePrevalenceRef(sampleSize = 1000)
cohort <- addCodelistAttribute(cohort = cdm$outcome, codelist = list("cohort_1" = 1L))
CDMConnector::cdm_disconnect(cdm)
```

## `codelistDiagnostics` *Run codelist-level diagnostics*

### Description

‘codelistDiagnostics()‘ runs phenotypeR diagnostics on the cohort_codelist attribute on the cohort. Thus codelist attribute of the cohort must be populated. If it is missing then it could be populated using ‘addCodelistAttribute()‘ function.

Furthermore ‘codelistDiagnostics()‘ requires achilles tables to be present in the cdm so that concept counts could be derived.

### Usage

```
codelistDiagnostics(cohort)
```

### Arguments

| | |
|---|---|
| `cohort` | A cohort table in a cdm reference. The cohort_codelist attribute must be populated. The cdm reference must contain achilles tables as these will be used for deriving concept counts. |

### Value

A summarised result

### Examples

```
  cdm_local <- omock::mockCdmReference() |>
  omock::mockPerson(nPerson = 100) |>
  omock::mockObservationPeriod() |>
  omock::mockConditionOccurrence() |>
  omock::mockDrugExposure() |>
  omock::mockCohort(name = "my_cohort")

  db <- DBI::dbConnect(duckdb::duckdb())

  cdm <- CDMConnector::copyCdmTo(con = db,
                                 cdm = cdm_local,
                                 schema ="main",
                                 overwrite = TRUE)

 result <- cdm$my_cohort |>
   codelistDiagnostics()

 CDMConnector::cdmDisconnect(cdm = cdm)
```

`cohortDiagnostics` *Run cohort-level diagnostics*

## Description

Runs phenotypeR diagnostics on the cohort. The diganostics include: * Age groups and sex summarised. * A summary of visits of everyone in the cohort using visit_occurrence table. * A summary of age and sex density of the cohort. * Attritions of the cohorts. * Overlap between cohorts (if more than one cohort is being used).

## Usage

```
cohortDiagnostics(cohort)
```

## Arguments

| | |
|---|---|
| `cohort` | Cohort table in a cdm reference |

## Value

A summarised result

## Examples

```
cdm_local <- omock::mockCdmReference() |>
  omock::mockPerson(nPerson = 100) |>
  omock::mockObservationPeriod() |>
  omock::mockConditionOccurrence() |>
  omock::mockDrugExposure() |>
  omock::mockObservation() |>
  omock::mockMeasurement() |>
  omock::mockVisitOccurrence() |>
  omock::mockProcedureOccurrence() |>
  omock::mockCohort(name = "my_cohort")
 db <- DBI::dbConnect(duckdb::duckdb())
 cdm <- CDMConnector::copyCdmTo(con = db,
                               cdm = cdm_local,
                               schema ="main",
                               overwrite = TRUE)

 cdm$my_cohort |> cohortDiagnostics()
 CDMConnector::cdmDisconnect(cdm = cdm)
```

## databaseDiagnostics *Database diagnostics*

### Description

phenotypeR diagnostics on the cdm object.

Diagnostics include: * Summarise a cdm_reference object, creating a snapshot with the metadata of the cdm_reference object. * Summarise the observation period table getting some overall statistics in a summarised_result object.

### Usage

```
databaseDiagnostics(cdm)
```

### Arguments

| | |
|---|---|
| `cdm` | CDM reference |

### Value

A summarised result

### Examples

```
 cdm_local <- omock::mockCdmReference() |>
   omock::mockPerson(nPerson = 100) |>
   omock::mockObservationPeriod() |>
   omock::mockConditionOccurrence() |>
   omock::mockDrugExposure() |>
   omock::mockObservation() |>
   omock::mockMeasurement() |>
   omock::mockCohort(name = "my_cohort", numberCohorts = 2)
db <- DBI::dbConnect(duckdb::duckdb())
cdm <- CDMConnector::copyCdmTo(con = db,
                               cdm = cdm_local,
                               schema ="main",
                               overwrite = TRUE)
db_diag <- databaseDiagnostics(cdm)
CDMConnector::cdmDisconnect(cdm = cdm)
```

## matchedDiagnostics *Compare characteristics of cohort matched to database population*

### Description

A summary of the cohort that is matched to the original cohort that has been given by the user. Such summary contains basic cohort summary including number of visits within one year prior of the cohort_start_date, as well as a large scale charactersitics using the following domians of OMOP CDM:

* condition_occurrence * visit_occurrence * measurement * procedure_occurrence * observation * drug_exposure

### Usage

```
matchedDiagnostics(cohort, matchedSample = 1000)
```

### Arguments

| | |
|---|---|
| `cohort` | Cohort table in a cdm reference |
| `matchedSample` | The number of people to take a random sample for matching. If NULL, no sampling will be performed. |

### Value

A summarised result

### Examples

```
cdm_local <- omock::mockCdmReference() |>
  omock::mockPerson(nPerson = 100) |>
  omock::mockObservationPeriod() |>
  omock::mockConditionOccurrence() |>
  omock::mockDrugExposure() |>
  omock::mockObservation() |>
  omock::mockMeasurement() |>
  omock::mockVisitOccurrence() |>
  omock::mockProcedureOccurrence() |>
  omock::mockCohort(name = "my_cohort")

 db <- DBI::dbConnect(duckdb::duckdb())
 cdm <- CDMConnector::copyCdmTo(con = db,
                               cdm = cdm_local,
                               schema ="main",
                               overwrite = TRUE)
 result <- cdm$my_cohort |> matchedDiagnostics()
 CDMConnector::cdm_disconnect(cdm)
```

## `phenotypeDiagnostics` *Phenotype a cohort*

### Description

This comprises all the diagnostics that are being offered in this package, this includes:

* A diagnostics on the database via ‘databaseDiagnostics‘. * A diagnostics on the cohort_codelist attribute of the cohort via ‘codelistDiagnostics‘. * A diagnostics on the cohort via ‘cohortDiagnostics‘. * A diagnostics on the population via ‘populationDiagnostics‘. * A diagnostics on the matched cohort via ‘matchedDiagnostics‘.

### Usage

```
phenotypeDiagnostics(
  cohort,
  databaseDiagnostics = TRUE,
  codelistDiagnostics = TRUE,
  cohortDiagnostics = TRUE,
  populationDiagnostics = TRUE,
  populationSample = 1e+06,
  populationDateRange = as.Date(c(NA, NA)),
  matchedDiagnostics = TRUE,
  matchedSample = 1000
)
```

### Arguments

| | |
|---|---|
| `cohort` | Cohort table in a cdm reference |
| `databaseDiagnostics` | If TRUE, database diagnostics will be run. |
| `codelistDiagnostics` | If TRUE, codelist diagnostics will be run. |
| `cohortDiagnostics` | If TRUE, cohort diagnostics will be run. |
| `populationDiagnostics` | If TRUE, population diagnostics will be run. |
| `populationSample` | Number of people from the cdm to sample. If NULL no sampling will be performed |
| `populationDateRange` | Two dates. The first indicating the earliest cohort start date and the second indicating the latest possible cohort end date. If NULL or the first date is set as missing, the earliest observation_start_date in the observation_period table will be used for the former. If NULL or the second date is set as missing, the latest observation_end_date in the observation_period table will be used for the latter. |

| | |
|---|---|
| `matchedDiagnostics` | If TRUE, cohort to population diagnostics will be run. |
| `matchedSample` | The number of people to take a random sample for matching. If NULL, no sampling will be performed. |

### Value

A summarised result

### Examples

```
cdm_local <- omock::mockCdmReference() |>
  omock::mockPerson(nPerson = 100) |>
  omock::mockObservationPeriod() |>
  omock::mockConditionOccurrence() |>
  omock::mockDrugExposure() |>
  omock::mockObservation() |>
  omock::mockMeasurement() |>
  omock::mockVisitOccurrence() |>
  omock::mockProcedureOccurrence() |>
  omock::mockCohort(name = "my_cohort")

  db <- DBI::dbConnect(duckdb::duckdb())
  cdm <- CDMConnector::copyCdmTo(con = db,
                                 cdm = cdm_local,
                                 schema ="main",
                                 overwrite = TRUE)
  phenotypeDiagnostics(cdm$my_cohort)
  CDMConnector::cdm_disconnect(cdm)
```

## populationDiagnostics *Population-level diagnostics*

### Description

phenotypeR diagnostics on the cohort of input with relation to a denomination population. Diagnostics include:

* Incidence * Prevalence

### Usage

```
populationDiagnostics(
  cohort,
  populationSample = 1e+06,
  populationDateRange = as.Date(c(NA, NA))
)
```

### Arguments

| | |
|---|---|
| `cohort` | Cohort table in a cdm reference |
| `populationSample` | Number of people from the cdm to sample. If NULL no sampling will be performed |
| `populationDateRange` | Two dates. The first indicating the earliest cohort start date and the second indicating the latest possible cohort end date. If NULL or the first date is set as missing, the earliest observation_start_date in the observation_period table will be used for the former. If NULL or the second date is set as missing, the latest observation_end_date in the observation_period table will be used for the latter. |

### Value

A summarised result

### Examples

```
library(IncidencePrevalence)
cdm <- mockIncidencePrevalenceRef(sampleSize = 1000)
pop_diag <- populationDiagnostics(cohort = cdm$outcome,
                                  populationSample = 250)
CDMConnector::cdm_disconnect(cdm)
```

## `shinyDiagnostics` *Create a shiny app summarising your phenotyping results*

### Description

A shiny app that is designed for any diagnostics results from phenotypeR, this includes:

* A diagnostics on the database via ‘databaseDiagnostics‘. * A diagnostics on the cohort_codelist attribute of the cohort via ‘codelistDiagnostics‘. * A diagnostics on the cohort via ‘cohortDiagnostics‘. * A diagnostics on the population via ‘populationDiagnostics‘. * A diagnostics on the matched cohort via ‘matchedDiagnostics‘.

### Usage

```
shinyDiagnostics(result, directory, open = rlang::is_interactive())
```

### Arguments

| | |
|---|---|
| `result` | A summarised result |
| `directory` | Directory where to save report |
| `open` | If TRUE, the shiny app will be launched in a new session. If FALSE, the shiny app will be created but not launched. |

## Value

A shiny app

## Examples

```
cdm_local <- omock::mockCdmReference() |>
  omock::mockPerson(nPerson = 100) |>
  omock::mockObservationPeriod() |>
  omock::mockConditionOccurrence() |>
  omock::mockDrugExposure() |>
  omock::mockObservation() |>
  omock::mockMeasurement() |>
  omock::mockVisitOccurrence() |>
  omock::mockProcedureOccurrence() |>
  omock::mockCohort(name = "my_cohort")

  db <- DBI::dbConnect(duckdb::duckdb())
  cdm <- CDMConnector::copyCdmTo(con = db,
                                 cdm = cdm_local,
                                 schema ="main",
                                 overwrite = TRUE)
  my_result_cohort_diag <- cdm$my_cohort |> phenotypeDiagnostics()
  shinyDiagnostics(my_result_cohort_diag, tempdir())
```

# Index