Package: DrugUtilisation (via r-universe)

July 3, 2026

Title Summarise Patient-Level Drug Utilisation in Data Mapped to the OMOP Common Data Model

Version 1.2.1

Description Summarise patient-level drug utilisation cohorts using data mapped to the Observational Medical Outcomes Partnership (OMOP) common data model. New users and prevalent users cohorts can be generated and their characteristics, indication and drug use summarised.

License Apache License (>= 2)

Encoding UTF-8

RoxygenNote 7.3.3

Suggests bit64, CDMConnector (>= 1.4.0), CohortConstructor, CohortSurvival, covr, DBI, downlit, duckdb, extrafont, flextable, ggplot2, ggtext, gt, here, knitr, odbc, omock, plotly, rmarkdown, RPostgres, scales, testthat (>= 3.1.5), tibble, visOmopResults (>= 1.3.0), xml2

Config/testthat/edition 3

Imports cli, clock, CodelistGenerator (>= 3.1.0), dplyr, glue, lifecycle, omopgenerics (>= 1.3.1), PatientProfiles (>= 1.4.2), purrr, rlang, stringr, tidyr

Depends R (>= 4.1)

LazyData true

URL https://darwin-eu.github.io/DrugUtilisation/

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

Config/testthat/parallel true

VignetteBuilder knitr

NeedsCompilation no

Author Martí Català [aut, cre] (ORCID: <https://orcid.org/0000-0003-3308-9905>), Mike Du [ctb] (ORCID: <https://orcid.org/0000-0002-9517-8834>), Yuchen Guo [aut]

(ORCID: <https://orcid.org/0000-0002-0847-4855>), Kim Lopez-Guell [aut] (ORCID: <https://orcid.org/0000-0002-8462-8668>), Edward Burn [aut] (ORCID: <https://orcid.org/0000-0002-9286-1128>), Xintong Li [ctb] (ORCID: <https://orcid.org/0000-0002-6872-5804>), Marta Alcalde-Herraiz [ctb] (ORCID: <https://orcid.org/0009-0002-4405-1814>), Nuria Mercade-Besora [aut] (ORCID: <https://orcid.org/0009-0006-7948-3747>), Xihang Chen [aut] (ORCID: <https://orcid.org/0009-0001-8112-8959>)

Maintainer Martí Català <marti.catalasabate@ndorms.ox.ac.uk>

Repository https://cran.r-universe.dev

Date/Publication 2026-07-03 02:50:13 UTC

RemoteUrl https://github.com/cran/DrugUtilisation

RemoteRef HEAD

RemoteSha 49fd31bb885397d89d2b13f834847b50d88050a7

Contents

`addCumulativeDose` *To add a new column with the cumulative dose. To add multiple columns use* `addDrugUtilisation()` *for efficiency.*

Description

To add a new column with the cumulative dose. To add multiple columns use `addDrugUtilisation()` for efficiency.

Usage

```
addCumulativeDose(
  cohort,
  ingredientConceptId,
  conceptSet = NULL,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  nameStyle = "cumulative_dose_{concept_name}_{ingredient}",
  name = NULL
)
```

Arguments

cohort	A cohort_table object.
ingredientConceptId	Ingredient OMOP concept that we are interested for the study.
conceptSet	List of concepts to be included.
indexDate	Name of a column that indicates the date to start the analysis.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
restrictIncident	Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.
nameStyle	Character string to specify the nameStyle of the new columns.
name	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The same cohort with the added column.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)

cdm$dus_cohort |>
  addCumulativeDose(ingredientConceptId = 1125315) |>
  glimpse()
```

---

addCumulativeQuantity *To add a new column with the cumulative quantity. To add multiple columns use* addDrugUtilisation() *for efficiency.*

Description

To add a new column with the cumulative quantity. To add multiple columns use addDrugUtilisation() for efficiency.

Usage

```
addCumulativeQuantity(
  cohort,
  conceptSet,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  nameStyle = "cumulative_quantity_{concept_name}",
  name = NULL
)
```

Arguments

cohort	A cohort_table object.
conceptSet	List of concepts to be included.
indexDate	Name of a column that indicates the date to start the analysis.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
restrictIncident	Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.
nameStyle	Character string to specify the nameStyle of the new columns.
name	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The same cohort with the added column.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)

cdm$dus_cohort |>
  addCumulativeQuantity(conceptSet = codelist) |>
  glimpse()
```

addDailyDose *Add daily dose to* drug_exposure *like table*

Description

Add daily dose to drug_exposure like table

Usage

```
addDailyDose(x, ingredientConceptId, name = NULL)
```

Arguments

x	A cdm_table class table with at least drug_concept_id, drug_exposure_start_date, drug_exposure_end_date and quantity as columns.
ingredientConceptId	Ingredient OMOP concept that we are interested for the study.
name	Name of the new generated table, if NULL a temporary table will be generated.

Value

A cdm_table with two new columns 'daily_dose' and 'unit'.

addDaysExposed *To add a new column with the days exposed. To add multiple columns use* addDrugUtilisation() *for efficiency.*

Description

To add a new column with the days exposed. To add multiple columns use addDrugUtilisation() for efficiency.

Usage

```
addDaysExposed(
  cohort,
  conceptSet,
  gapEra,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  nameStyle = "days_exposed_{concept_name}",
  name = NULL
)
```

Arguments

cohort	A cohort_table object.
conceptSet	List of concepts to be included.
gapEra	Number of days between two continuous exposures to be considered in the same era.
indexDate	Name of a column that indicates the date to start the analysis.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
restrictIncident	Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.
nameStyle	Character string to specify the nameStyle of the new columns.
name	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The same cohort with the added column.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)

cdm$dus_cohort |>
  addDaysExposed(conceptSet = codelist, gapEra = 1) |>
  glimpse()
```

---

addDaysPrescribed *To add a new column with the days prescribed. To add multiple columns use* addDrugUtilisation() *for efficiency.*

Description

To add a new column with the days prescribed. To add multiple columns use addDrugUtilisation() for efficiency.

Usage

```
addDaysPrescribed(
  cohort,
  conceptSet,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  nameStyle = "days_prescribed_{concept_name}",
  name = NULL
)
```

Arguments

cohort	A cohort_table object.
conceptSet	List of concepts to be included.
indexDate	Name of a column that indicates the date to start the analysis.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
restrictIncident	Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.
nameStyle	Character string to specify the nameStyle of the new columns.
name	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The same cohort with the added columns.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)

cdm$dus_cohort |>
  addDaysPrescribed(conceptSet = codelist) |>
  glimpse()
```

addDrugRestart *Add drug restart information as a column per follow-up period of interest.*

Description

Add drug restart information as a column per follow-up period of interest.

Usage

```
addDrugRestart(
  cohort,
  switchCohortTable,
  switchCohortId = NULL,
  followUpDays = Inf,
  censorDate = NULL,
  incident = TRUE,
  nameStyle = "drug_restart_{follow_up_days}"
)
```

Arguments

cohort	A cohort_table object.
switchCohortTable	A cohort table in the cdm that contains possible alternative treatments.
switchCohortId	The cohort ids to be used from switchCohortTable. If NULL all cohort definition ids are used.
followUpDays	A vector of number of days to follow up. It can be multiple values.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
incident	Whether the switch treatment has to be incident (start after discontinuation) or not (it can start before the discontinuation and last till after).
nameStyle	Character string to specify the nameStyle of the new columns.

Value

The cohort table given with additional columns with information on the restart, switch and not exposed per follow-up period of interest.

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

conceptlist <- list(acetaminophen = 1125360, metformin = c(1503297, 1503327))
```

```
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "switch_cohort",
                                        conceptSet = conceptlist)

cdm$cohort1 |>
  addDrugRestart(switchCohortTable = "switch_cohort")
```

`addDrugUtilisation` *Add new columns with drug use related information*

Description

Add new columns with drug use related information

Usage

```
addDrugUtilisation(
  cohort,
  gapEra,
  conceptSet = NULL,
  ingredientConceptId = NULL,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  numberExposures = TRUE,
  numberEras = TRUE,
  daysExposed = TRUE,
  daysPrescribed = TRUE,
  timeToExposure = TRUE,
  initialExposureDuration = TRUE,
  initialQuantity = TRUE,
  cumulativeQuantity = TRUE,
  initialDailyDose = TRUE,
  cumulativeDose = TRUE,
  nameStyle = "{value}_{concept_name}_{ingredient}",
  name = NULL
)
```

Arguments

`cohort`	A cohort_table object.
`gapEra`	Number of days between two continuous exposures to be considered in the same era.
`conceptSet`	List of concepts to be included.
`ingredientConceptId`	Ingredient OMOP concept that we are interested for the study.

indexDate	Name of a column that indicates the date to start the analysis.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
restrictIncident	Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.
numberExposures	Whether to include 'number_exposures' (number of drug exposure records between indexDate and censorDate).
numberEras	Whether to include 'number_eras' (number of continuous exposure episodes between indexDate and censorDate).
daysExposed	Whether to include 'days_exposed' (number of days that the individual is in a continuous exposure episode, including allowed treatment gaps, between indexDate and censorDate; sum of the length of the different drug eras).
daysPrescribed	Whether to include 'days_prescribed' (sum of the number of days for each prescription that contribute in the analysis).
timeToExposure	Whether to include 'time_to_exposure' (number of days between indexDate and the first episode).
initialExposureDuration	Whether to include 'initial_exposure_duration' (number of prescribed days of the first drug exposure record).
initialQuantity	Whether to include 'initial_quantity' (quantity of the first drug exposure record).
cumulativeQuantity	Whether to include 'cumulative_quantity' (sum of the quantity of the different exposures considered in the analysis).
initialDailyDose	Whether to include 'initial_daily_dose_{unit}' (daily dose of the first considered prescription).
cumulativeDose	Whether to include 'cumulative_dose_{unit}' (sum of the cumulative dose of the analysed drug exposure records).
nameStyle	Character string to specify the nameStyle of the new columns.
name	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The same cohort with the added columns.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()
```

```
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")

cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)
cdm$dus_cohort |>
  addDrugUtilisation(ingredientConceptId = 1125315, gapEra = 30) |>
  glimpse()
```

`addIndication` *Add a variable indicating individuals indications*

Description

Add a variable to a drug cohort indicating their presence in an indication cohort in a specified time window. If an individual is not in one of the indication cohorts, they will be considered to have an unknown indication if they are present in one of the specified OMOP CDM clinical tables. If they are neither in an indication cohort or a clinical table they will be considered as having no observed indication.

Usage

```
addIndication(
  cohort,
  indicationCohortName,
  indicationCohortId = NULL,
  indicationWindow = list(c(0, 0)),
  unknownIndicationTable = NULL,
  indexDate = "cohort_start_date",
  censorDate = NULL,
  mutuallyExclusive = TRUE,
  nameStyle = NULL,
  name = NULL
)
```

Arguments

`cohort`	A cohort_table object.
`indicationCohortName`	Name of indication cohort table
`indicationCohortId`	target cohort Id to add indication
`indicationWindow`	time window of interests
`unknownIndicationTable`	Tables to search unknown indications

indexDate	Name of a column that indicates the date to start the analysis.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
mutuallyExclusive	Whether to consider mutually exclusive categories (one column per window) or not (one column per window and indication).
nameStyle	Name style for the indications. By default: 'indication_{window_name}' (mutuallyExclusive = TRUE), 'indication_{window_name}_{cohort_name}' (mutuallyExclusive = FALSE).
name	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The original table with a variable added that summarises the individual´s indications.

Examples

```
library(DrugUtilisation)
library(dplyr, warn.conflicts = FALSE)
library(CDMConnector)

cdm <- mockDrugUtilisation(source = "duckdb")

indications <- list(headache = 378253, asthma = 317009)
cdm <- generateConceptCohortSet(cdm = cdm,
                                conceptSet = indications,
                                name = "indication_cohorts")

cdm <- generateIngredientCohortSet(cdm = cdm,
                                   name = "drug_cohort",
                                   ingredient = "acetaminophen")

cdm$drug_cohort |>
  addIndication(
    indicationCohortName = "indication_cohorts",
    indicationWindow = list(c(0, 0)),
    unknownIndicationTable = "condition_occurrence"
  ) |>
  glimpse()
```

`addInitialDailyDose` *To add a new column with the initial daily dose. To add multiple columns use* `addDrugUtilisation()` *for efficiency.*

Description

To add a new column with the initial daily dose. To add multiple columns use `addDrugUtilisation()` for efficiency.

Usage

```
addInitialDailyDose(
  cohort,
  ingredientConceptId,
  conceptSet = NULL,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  nameStyle = "initial_daily_dose_{concept_name}_{ingredient}",
  name = NULL
)
```

Arguments

`cohort`	A cohort_table object.
`ingredientConceptId`	Ingredient OMOP concept that we are interested for the study.
`conceptSet`	List of concepts to be included.
`indexDate`	Name of a column that indicates the date to start the analysis.
`censorDate`	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
`restrictIncident`	Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.
`nameStyle`	Character string to specify the nameStyle of the new columns.
`name`	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The same cohort with the added column.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)
```

```
cdm$dus_cohort |>
  addInitialDailyDose(ingredientConceptId = 1125315) |>
  glimpse()
```

`addInitialExposureDuration` *To add a new column with the duration of the first exposure. To add multiple columns use* `addDrugUtilisation()` *for efficiency.*

Description

To add a new column with the duration of the first exposure. To add multiple columns use `addDrugUtilisation()` for efficiency.

Usage

```
addInitialExposureDuration(
  cohort,
  conceptSet,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  nameStyle = "initial_exposure_duration_{concept_name}",
  name = NULL
)
```

Arguments

`cohort`	A cohort_table object.
`conceptSet`	List of concepts to be included.
`indexDate`	Name of a column that indicates the date to start the analysis.
`censorDate`	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
`restrictIncident`	Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.
`nameStyle`	Character string to specify the nameStyle of the new columns.
`name`	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The same cohort with the added column.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)

cdm$dus_cohort |>
  addInitialExposureDuration(conceptSet = codelist) |>
  glimpse()
```

---

addInitialQuantity *To add a new column with the initial quantity. To add multiple columns use* addDrugUtilisation() *for efficiency.*

Description

To add a new column with the initial quantity. To add multiple columns use addDrugUtilisation() for efficiency.

Usage

```
addInitialQuantity(
  cohort,
  conceptSet,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  nameStyle = "initial_quantity_{concept_name}",
  name = NULL
)
```

Arguments

cohort	A cohort_table object.
conceptSet	List of concepts to be included.
indexDate	Name of a column that indicates the date to start the analysis.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
restrictIncident	Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.

nameStyle	Character string to specify the nameStyle of the new columns.
name	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The same cohort with the added column.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)

cdm$dus_cohort |>
  addInitialQuantity(conceptSet = codelist) |>
  glimpse()
```

---

addNumberEras

To add a new column with the number of eras. To add multiple columns use addDrugUtilisation() *for efficiency.*

Description

To add a new column with the number of eras. To add multiple columns use addDrugUtilisation() for efficiency.

Usage

```
addNumberEras(
  cohort,
  conceptSet,
  gapEra,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  nameStyle = "number_eras_{concept_name}",
  name = NULL
)
```

Arguments

cohort	A cohort_table object.
conceptSet	List of concepts to be included.
gapEra	Number of days between two continuous exposures to be considered in the same era.
indexDate	Name of a column that indicates the date to start the analysis.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
restrictIncident	Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.
nameStyle	Character string to specify the nameStyle of the new columns.
name	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The same cohort with the added column.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)

cdm$dus_cohort |>
  addNumberEras(conceptSet = codelist, gapEra = 1) |>
  glimpse()
```

---

addNumberExposures *To add a new column with the number of exposures. To add multiple columns use* addDrugUtilisation() *for efficiency.*

Description

To add a new column with the number of exposures. To add multiple columns use addDrugUtilisation() for efficiency.

Usage

```
addNumberExposures(
  cohort,
  conceptSet,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  nameStyle = "number_exposures_{concept_name}",
  name = NULL
)
```

Arguments

cohort	A cohort_table object.
conceptSet	List of concepts to be included.
indexDate	Name of a column that indicates the date to start the analysis.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
restrictIncident	Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.
nameStyle	Character string to specify the nameStyle of the new columns.
name	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The same cohort with the added columns.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)

cdm$dus_cohort |>
  addNumberExposures(conceptSet = codelist) |>
  glimpse()
```

addTimeToExposure *To add a new column with the time to exposure. To add multiple columns use* addDrugUtilisation() *for efficiency.*

Description

To add a new column with the time to exposure. To add multiple columns use addDrugUtilisation() for efficiency.

Usage

```
addTimeToExposure(
  cohort,
  conceptSet,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  nameStyle = "time_to_exposure_{concept_name}",
  name = NULL
)
```

Arguments

cohort	A cohort_table object.
conceptSet	List of concepts to be included.
indexDate	Name of a column that indicates the date to start the analysis.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
restrictIncident	Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.
nameStyle	Character string to specify the nameStyle of the new columns.
name	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The same cohort with the added column.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()
```

```
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)

cdm$dus_cohort |>
  addTimeToExposure(conceptSet = codelist) |>
  glimpse()
```

`addTreatment` *Add a variable indicating individuals medications*

Description

Add a variable to a drug cohort indicating their presence of a medication cohort in a specified time window.

Usage

```
addTreatment(
  cohort,
  treatmentCohortName,
  treatmentCohortId = NULL,
  window = list(c(0, 0)),
  indexDate = "cohort_start_date",
  censorDate = NULL,
  mutuallyExclusive = TRUE,
  nameStyle = NULL,
  name = NULL
)
```

Arguments

`cohort`	A cohort_table object.
`treatmentCohortName`	Name of treatment cohort table
`treatmentCohortId`	target cohort Id to add treatment
`window`	time window of interests.
`indexDate`	Name of a column that indicates the date to start the analysis.
`censorDate`	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
`mutuallyExclusive`	Whether to consider mutually exclusive categories (one column per window) or not (one column per window and treatment).

nameStyle	Name style for the treatment columns. By default: 'treatment_{window_name}' (mutuallyExclusive = TRUE), 'treatment_{window_name}_{cohort_name}' (mutuallyExclusive = FALSE).
name	Name of the new computed cohort table, if NULL a temporary table will be created.

Value

The original table with a variable added that summarises the individual´s indications.

Examples

```
library(DrugUtilisation)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation(numberIndividuals = 50)

cdm <- generateIngredientCohortSet(cdm = cdm,
                                   name = "drug_cohort",
                                   ingredient = "acetaminophen")

cdm <- generateIngredientCohortSet(cdm = cdm,
                                   name = "treatments",
                                   ingredient = c("metformin", "simvastatin"))

cdm$drug_cohort |>
  addTreatment("treatments", window = list(c(0, 0), c(1, 30), c(31, 60))) |>
  glimpse()
```

---

benchmarkDrugUtilisation

Run benchmark of drug utilisation cohort generation

Description

Run benchmark of drug utilisation cohort generation

Usage

```
benchmarkDrugUtilisation(
  cdm,
  ingredient = "acetaminophen",
  alternativeIngredient = c("ibuprofen", "aspirin", "diclofenac"),
  indicationCohort = NULL
)
```

Arguments

cdm	A cdm_reference object.
ingredient	Name of ingredient to benchmark.
alternativeIngredient	Name of ingredients to use as alternative treatments.
indicationCohort	Name of a cohort in the cdm_reference object to use as indicatiomn.

Value

A summarise_result object.

Examples

```
library(DrugUtilisation)
library(omock)

cdm <- mockCdmFromDataset(datasetName = "GiBleed", source = "duckdb")

timings <- benchmarkDrugUtilisation(cdm)

timings
```

---

cohortGapEra *Get the gapEra used to create a cohort*

Description

Get the gapEra used to create a cohort

Usage

```
cohortGapEra(cohort, cohortId = NULL)
```

Arguments

cohort	A cohort_table object.
cohortId	Integer vector refering to cohortIds from cohort. If NULL all cohort definition ids in settings will be used.

Value

gapEra values for the specific cohortIds

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)

cdm <- mockDrugUtilisation()

druglist <- getDrugIngredientCodes(cdm = cdm,
                                   name = c("acetaminophen", "metformin"))

cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "drug_cohorts",
                                        conceptSet = druglist,
                                        gapEra = 100)

cohortGapEra(cdm$drug_cohorts)
```

`erafyCohort` *Erafy a cohort_table collapsing records separated gapEra days or less.*

Description

Erafy a cohort_table collapsing records separated gapEra days or less.

Usage

```
erafyCohort(
  cohort,
  gapEra,
  cohortId = NULL,
  nameStyle = "{cohort_name}_{gap_era}",
  name = omopgenerics::tableName(cohort)
)
```

Arguments

`cohort`	A cohort_table object.
`gapEra`	Number of days between two continuous exposures to be considered in the same era.
`cohortId`	A cohort definition id to restrict by. If NULL, all cohorts will be included.
`nameStyle`	String to create the new names of cohorts. Must contain '{cohort_name}' if more than one cohort is present and '{gap_era}' if more than one gapEra is provided.
`name`	Name of the new cohort table, it must be a length 1 character vector.

Value

A cohort_table object.

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

cdm$cohort2 <- cdm$cohort1 |>
  erafyCohort(gapEra = 30, name = "cohort2")

cdm$cohort2

settings(cdm$cohort2)
```

---

generateAtcCohortSet *Generate a set of drug cohorts based on ATC classification*

Description

Adds a new cohort table to the cdm reference with individuals who have drug exposure records that belong to the specified Anatomical Therapeutic Chemical (ATC) classification. Cohort start and end dates will be based on drug record start and end dates, respectively. Records that overlap or have fewer days between them than the specified gap era will be concatenated into a single cohort entry.

Usage

```
generateAtcCohortSet(
  cdm,
  name,
  atcName = NULL,
  gapEra = 1,
  subsetCohort = NULL,
  subsetCohortId = NULL,
  numberExposures = FALSE,
  daysPrescribed = FALSE,
  ...
)
```

Arguments

cdm	A cdm_reference object.
name	Name of the new cohort table, it must be a length 1 character vector.

atcName	Names of ATC classification of interest.
gapEra	Number of days between two continuous exposures to be considered in the same era. Records that have fewer days between them than this gap will be concatenated into the same cohort record.
subsetCohort	Cohort table to subset.
subsetCohortId	Cohort id to subset.
numberExposures	Whether to include 'number_exposures' (number of drug exposure records between indexDate and censorDate).
daysPrescribed	Whether to include 'days_prescribed' (number of days prescribed used to create each era).
...	Arguments to be passed to `CodelistGenerator::getATCCodes()`.

Value

The function returns the cdm reference provided with the addition of the new cohort table.

Examples

```
library(DrugUtilisation)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()

cdm <- generateAtcCohortSet(cdm = cdm,
                            atcName = "alimentary tract and metabolism",
                            name = "drugs")

cdm$drugs |>
  glimpse()
```

---

generateDrugUtilisationCohortSet

Generate a set of drug cohorts based on given concepts

Description

Adds a new cohort table to the cdm reference with individuals who have drug exposure records with the specified concepts. Cohort start and end dates will be based on drug record start and end dates, respectively. Records that overlap or have fewer days between them than the specified gap era will be concatenated into a single cohort entry.

Usage

```
generateDrugUtilisationCohortSet(
  cdm,
  name,
  conceptSet,
  gapEra = 1,
  subsetCohort = NULL,
  subsetCohortId = NULL,
  numberExposures = FALSE,
  daysPrescribed = FALSE
)
```

Arguments

cdm	A cdm_reference object.
name	Name of the new cohort table, it must be a length 1 character vector.
conceptSet	List of concepts to be included.
gapEra	Number of days between two continuous exposures to be considered in the same era.
subsetCohort	Cohort table to subset.
subsetCohortId	Cohort id to subset.
numberExposures	Whether to include 'number_exposures' (number of drug exposure records between indexDate and censorDate).
daysPrescribed	Whether to include 'days_prescribed' (number of days prescribed used to create each era).

Value

The function returns the cdm reference provided with the addition of the new cohort table.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()

druglist <- getDrugIngredientCodes(cdm = cdm,
                                   name = c("acetaminophen", "metformin"),
                                   nameStyle = "{concept_name}")

cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "drug_cohorts",
                                        conceptSet = druglist,
                                        gapEra = 30,
                                        numberExposures = TRUE,
```

```
                                        daysPrescribed = TRUE)

cdm$drug_cohorts |>
  glimpse()
```

`generateIngredientCohortSet` *Generate a set of drug cohorts based on drug ingredients*

Description

Adds a new cohort table to the cdm reference with individuals who have drug exposure records with the specified drug ingredient. Cohort start and end dates will be based on drug record start and end dates, respectively. Records that overlap or have fewer days between them than the specified gap era will be concatenated into a single cohort entry.

Usage

```
generateIngredientCohortSet(
  cdm,
  name,
  ingredient = NULL,
  gapEra = 1,
  subsetCohort = NULL,
  subsetCohortId = NULL,
  numberExposures = FALSE,
  daysPrescribed = FALSE,
  ...
)
```

Arguments

`cdm`	A `cdm_reference` object.
`name`	Name of the new cohort table, it must be a length 1 character vector.
`ingredient`	Accepts both vectors and named lists of ingredient names. For a vector input, e.g., c("acetaminophen", "codeine"), it generates a cohort table with descendant concept codes for each ingredient, assigning unique cohort_definition_id. For a named list input, e.g., list("test_1" = c("simvastatin", "acetaminophen"), "test_2" = "metformin"), it produces a cohort table based on the structure of the input, where each name leads to a combined set of descendant concept codes for the specified ingredients, creating distinct cohort_definition_id for each named group.
`gapEra`	Number of days between two continuous exposures to be considered in the same era.

subsetCohort	Cohort table to subset.
subsetCohortId	Cohort id to subset.
numberExposures	Whether to include 'number_exposures' (number of drug exposure records between indexDate and censorDate).
daysPrescribed	Whether to include 'days_prescribed' (number of days prescribed used to create each era).
...	Arguments to be passed to `CodelistGenerator::getDrugIngredientCodes()`.

Value

The function returns the cdm reference provided with the addition of the new cohort table.

Examples

```
library(DrugUtilisation)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()

cdm <- generateIngredientCohortSet(cdm = cdm,
                                   ingredient = "acetaminophen",
                                   name = "acetaminophen")

cdm$acetaminophen |>
  glimpse()
```

---

mockDrugUtilisation *It creates a mock database for testing DrugUtilisation package*

Description

It creates a mock database for testing DrugUtilisation package

Usage

```
mockDrugUtilisation(
  numberIndividuals = 10,
  ...,
  source = "local",
  con = lifecycle::deprecated(),
  writeSchema = lifecycle::deprecated(),
  seed = lifecycle::deprecated()
)
```

Arguments

numberIndividuals	Number of individuals in the mock cdm.
...	Tables to use as basis to create the mock. If some tables are provided they will be used to construct the cdm object.
source	Source for the mock cdm, it can either be 'local' or 'duckdb'.
con	deprecated.
writeSchema	deprecated.
seed	deprecated.

Value

A cdm reference with the mock tables

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

cdm
```

---

patternsWithFormula *Patterns valid to compute daily dose with the associated formula.*

Description

Patterns valid to compute daily dose with the associated formula.

Usage

```
patternsWithFormula
```

Format

A data frame with eight variables: `pattern_id`, `amount`, `amount_unit`, `numerator`, `numerator_unit`, `denominator`, `denominator_unit`, `formula_name` and `formula`.

patternTable *Function to create a tibble with the patterns from current drug strength table*

Description

Function to create a tibble with the patterns from current drug strength table

Usage

```
patternTable(cdm)
```

Arguments

cdm	A cdm_reference object.

Value

The function creates a tibble with the different patterns found in the table, plus a column of potentially valid and invalid combinations.

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

patternTable(cdm)
```

plotDiscontinuationAsSurvival *Plot discontinuation*

Description

Plot discontinuation

Usage

```
plotDiscontinuationAsSurvival(
  result,
  facet = "cohort_name",
  colour = c("variable_name", strataColumns(result)),
  ribbon = TRUE,
  style = NULL
)
```

Arguments

result	A summarised_result object.
facet	Columns to facet by. See options with availablePlotColumns(result). Formula is also allowed to specify rows and columns.
colour	Columns to color by. See options with availablePlotColumns(result).
ribbon	Whether to plot a ribbon with the confidence intervals.
style	Visual theme to apply. Character, or NULL. If a character, this may be either the name of a built-in style (see plotStyle()), or a path to a .yml file that defines a custom style. If NULL, the function will use the explicit default style, unless a global style option is set (see visOmopResults::setGlobalPlotOptions()) or a _brand.yml file is present (in that order). Refer to the **visOmopResults** package vignette on styles to learn more.

Value

Plot probability to continue the drug over over time

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

result <- cdm$cohort1 |>
  summariseDiscontinuationAsSurvival(followUpDays = 365)

plotDiscontinuationAsSurvival(result)
```

---

plotDrugRestart *Generate a custom ggplot2 from a summarised_result object generated with summariseDrugRestart() function.*

Description

Generate a custom ggplot2 from a summarised_result object generated with summariseDrugRestart() function.

Usage

```
plotDrugRestart(
  result,
  x = "variable_level",
  position = "stack",
  facet = cdm_name + cohort_name ~ follow_up_days,
```

```
  colour = "variable_level",
  style = NULL
)
```

Arguments

result	A summarised_result object.
x	Variable to plot on x-axis
position	Position of bars, can be either dodge or stack
facet	Columns to facet by. See options with availablePlotColumns(result). Formula is also allowed to specify rows and columns.
colour	Columns to color by. See options with availablePlotColumns(result).
style	Visual theme to apply. Character, or NULL. If a character, this may be either the name of a built-in style (see plotStyle()), or a path to a .yml file that defines a custom style. If NULL, the function will use the explicit default style, unless a global style option is set (see visOmopResults::setGlobalPlotOptions()) or a _brand.yml file is present (in that order). Refer to the **visOmopResults** package vignette on styles to learn more.

Value

A ggplot2 object.

Examples

```
## Not run:
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

conceptlist <- list("a" = 1125360, "b" = c(1503297, 1503327))
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "switch_cohort",
                                        conceptSet = conceptlist)

result <- cdm$cohort1 |>
  summariseDrugRestart(switchCohortTable = "switch_cohort")

plotDrugRestart(result)

plotDrugRestart(result, x = "cohort_name", facet = "follow_up_days")

## End(Not run)
```

`plotDrugUtilisation` *Plot the results of* `summariseDrugUtilisation`

Description

Plot the results of `summariseDrugUtilisation`

Usage

```
plotDrugUtilisation(
  result,
  variable = "number exposures",
  plotType = "barplot",
  facet = strataColumns(result),
  colour = "cohort_name",
  style = NULL
)
```

Arguments

`result`	A summarised_result object.
`variable`	Variable to plot. See `unique(result$variable_name)` for options.
`plotType`	Must be a choice between: 'scatterplot', 'barplot', 'densityplot', and 'boxplot'.
`facet`	Columns to facet by. See options with `availablePlotColumns(result)`. Formula is also allowed to specify rows and columns.
`colour`	Columns to color by. See options with `availablePlotColumns(result)`.
`style`	Visual theme to apply. Character, or `NULL`. If a character, this may be either the name of a built-in style (see `plotStyle()`), or a path to a `.yml` file that defines a custom style. If NULL, the function will use the explicit default style, unless a global style option is set (see `visOmopResults::setGlobalPlotOptions()`) or a `_brand.yml` file is present (in that order). Refer to the **visOmopResults** package vignette on styles to learn more.

Value

A ggplot2 object.

Examples

```
library(DrugUtilisation)
library(PatientProfiles)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation(numberIndividuals = 100)
codes <- list(aceta = c(1125315, 1125360, 2905077, 43135274))
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
```

```
                                        name = "cohort",
                                        conceptSet = codes)

result <- cdm$cohort |>
  addSex() |>
  summariseDrugUtilisation(
    strata = "sex",
    ingredientConceptId = 1125315,
    estimates = c("min", "q25", "median", "q75", "max", "density")
  )

result |>
  filter(estimate_name == "median") |>
  plotDrugUtilisation(
    variable = "days prescribed",
    plotType = "barplot"
  )

result |>
  plotDrugUtilisation(
    variable = "days exposed",
    facet = cohort_name ~ cdm_name,
    colour = "sex",
    plotType = "boxplot"
  )

result |>
  plotDrugUtilisation(
    variable = "cumulative dose milligram",
    plotType = "densityplot",
    facet = "cohort_name",
    colour = "sex"
  )
```

`plotIndication` *Generate a plot visualisation (ggplot2) from the output of summariseIndication*

Description

Generate a plot visualisation (ggplot2) from the output of summariseIndication

Usage

```
plotIndication(
  result,
  x = "variable_level",
```

```
  position = "stack",
  facet = cdm_name + cohort_name ~ window_name,
  colour = "variable_level",
  style = NULL
)
```

Arguments

result	A summarised_result object.
x	Variable to plot on x-axis
position	Position of bars, can be either dodge or stack
facet	Columns to facet by. See options with availablePlotColumns(result). Formula is also allowed to specify rows and columns.
colour	Columns to color by. See options with availablePlotColumns(result).
style	Visual theme to apply. Character, or NULL. If a character, this may be either the name of a built-in style (see plotStyle()), or a path to a .yml file that defines a custom style. If NULL, the function will use the explicit default style, unless a global style option is set (see visOmopResults::setGlobalPlotOptions()) or a _brand.yml file is present (in that order). Refer to the **visOmopResults** package vignette on styles to learn more.

Value

A ggplot2 object

Examples

```
library(DrugUtilisation)
library(CDMConnector)

cdm <- mockDrugUtilisation(source = "duckdb")

indications <- list(headache = 378253, asthma = 317009)
cdm <- generateConceptCohortSet(cdm = cdm,
                                conceptSet = indications,
                                name = "indication_cohorts")

cdm <- generateIngredientCohortSet(cdm = cdm,
                                   name = "drug_cohort",
                                   ingredient = "acetaminophen")

result <- cdm$drug_cohort |>
  summariseIndication(
    indicationCohortName = "indication_cohorts",
    unknownIndicationTable = "condition_occurrence",
    indicationWindow = list(c(-Inf, 0), c(-365, 0))
  )

plotIndication(result)
```

```
plotIndication(result, x = "window_name", facet = NULL)
```

`plotProportionOfPatientsCovered` *Plot proportion of patients covered*

Description

Plot proportion of patients covered

Usage

```
plotProportionOfPatientsCovered(
  result,
  facet = "cohort_name",
  colour = strataColumns(result),
  ribbon = TRUE,
  style = NULL
)
```

Arguments

`result`	A summarised_result object.
`facet`	Columns to facet by. See options with `availablePlotColumns(result)`. Formula is also allowed to specify rows and columns.
`colour`	Columns to color by. See options with `availablePlotColumns(result)`.
`ribbon`	Whether to plot a ribbon with the confidence intervals.
`style`	Visual theme to apply. Character, or `NULL`. If a character, this may be either the name of a built-in style (see `plotStyle()`), or a path to a `.yml` file that defines a custom style. If NULL, the function will use the explicit default style, unless a global style option is set (see `visOmopResults::setGlobalPlotOptions()`) or a `_brand.yml` file is present (in that order). Refer to the **visOmopResults** package vignette on styles to learn more.

Value

Plot of proportion Of patients covered over time

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
```

```
                                        name = "my_cohort",
                                conceptSet = list(drug_of_interest = c(1503297, 1503327)))

result <- cdm$my_cohort |>
  summariseProportionOfPatientsCovered(followUpDays = 365)

plotProportionOfPatientsCovered(result)
```

`plotTreatment` *Generate a custom ggplot2 from a summarised_result object generated with summariseTreatment function.*

Description

Generate a custom ggplot2 from a summarised_result object generated with summariseTreatment function.

Usage

```
plotTreatment(
  result,
  x = "variable_level",
  position = "stack",
  facet = cdm_name + cohort_name ~ window_name,
  colour = "variable_level",
  style = NULL
)
```

Arguments

`result`	A summarised_result object.
`x`	Variable to plot on x-axis
`position`	Position of bars, can be either dodge or `stack`
`facet`	Columns to facet by. See options with `availablePlotColumns(result)`. Formula is also allowed to specify rows and columns.
`colour`	Columns to color by. See options with `availablePlotColumns(result)`.
`style`	Visual theme to apply. Character, or `NULL`. If a character, this may be either the name of a built-in style (see `plotStyle()`), or a path to a `.yml` file that defines a custom style. If NULL, the function will use the explicit default style, unless a global style option is set (see `visOmopResults::setGlobalPlotOptions()`) or a `_brand.yml` file is present (in that order). Refer to the **visOmopResults** package vignette on styles to learn more.

Value

A ggplot2 object.

Examples

```
## Not run:
library(DrugUtilisation)

cdm <- mockDrugUtilisation()
result <- cdm$cohort1 |>
  summariseTreatment(
    treatmentCohortName = "cohort2",
    window = list(c(0, 30), c(31, 365))
  )

plotTreatment(result)

plotTreatment(result, x = "cohort_name", facet = "window_name")

## End(Not run)
```

---

requireDrugInDateRange

Restrict cohort to only cohort records within a certain date range

---

Description

Filter the cohort table keeping only the cohort records for which the specified index date is within a specified date range.

Usage

```
requireDrugInDateRange(
  cohort,
  dateRange,
  indexDate = "cohort_start_date",
  cohortId = NULL,
  name = omopgenerics::tableName(cohort)
)
```

Arguments

cohort	A cohort_table object.
dateRange	Date interval to consider. Any records with the index date outside of this range will be dropped.
indexDate	The column containing the date that will be checked against the date range.
cohortId	A cohort definition id to restrict by. If NULL, all cohorts will be included.
name	Name of the new cohort table, it must be a length 1 character vector.

Value

The cohort table having applied the date requirement.

Examples

```
library(DrugUtilisation)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()

cdm$cohort1 <- cdm$cohort1 |>
  requireDrugInDateRange(dateRange = as.Date(c("2020-01-01", NA)))

attrition(cdm$cohort1) |>
  glimpse()
```

---

requireIsFirstDrugEntry

Restrict cohort to only the first cohort record per subject

Description

Filter the cohort table keeping only the first cohort record per subject.

Usage

```
requireIsFirstDrugEntry(
  cohort,
  cohortId = NULL,
  name = omopgenerics::tableName(cohort)
)
```

Arguments

cohort	A cohort_table object.
cohortId	A cohort definition id to restrict by. If NULL, all cohorts will be included.
name	Name of the new cohort table, it must be a length 1 character vector.

Value

The cohort table having applied the first entry requirement.

Examples

```
library(DrugUtilisation)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()

cdm$cohort1 <- cdm$cohort1 |>
  requireIsFirstDrugEntry()

attrition(cdm$cohort1) |>
  glimpse()
```

---

requireObservationBeforeDrug

Restrict cohort to only cohort records with the given amount of prior observation time in the database

---

Description

Filter the cohort table keeping only the cohort records for which the individual has the required observation time in the database prior to their cohort start date.

Usage

```
requireObservationBeforeDrug(
  cohort,
  days,
  cohortId = NULL,
  name = omopgenerics::tableName(cohort)
)
```

Arguments

cohort	A cohort_table object.
days	Number of days of prior observation required before cohort start date. Any records with fewer days will be dropped.
cohortId	A cohort definition id to restrict by. If NULL, all cohorts will be included.
name	Name of the new cohort table, it must be a length 1 character vector.

Value

The cohort table having applied the prior observation requirement.

Examples

```
library(DrugUtilisation)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()

cdm$cohort1 <- cdm$cohort1 |>
  requireObservationBeforeDrug(days = 365)

attrition(cdm$cohort1) |>
  glimpse()
```

---

requirePriorDrugWashout

Restrict cohort to only cohort records with a given amount of time since the last cohort record ended

---

Description

Filter the cohort table keeping only the cohort records for which the required amount of time has passed since the last cohort entry ended for that individual.

Usage

```
requirePriorDrugWashout(
  cohort,
  days,
  cohortId = NULL,
  name = omopgenerics::tableName(cohort)
)
```

Arguments

cohort	A cohort_table object.
days	The number of days required to have passed since the last cohort record finished. Any records with fewer days than this will be dropped. Note that setting days to Inf will lead to the same result as that from using the `requireIsFirstDrugEntry` function (with only an individual´s first cohort record kept).
cohortId	A cohort definition id to restrict by. If NULL, all cohorts will be included.
name	Name of the new cohort table, it must be a length 1 character vector.

Value

The cohort table having applied the washout requirement.

Examples

```
library(DrugUtilisation)
library(dplyr, warn.conflicts = FALSE)

cdm <- mockDrugUtilisation()

cdm$cohort1 <- cdm$cohort1 |>
  requirePriorDrugWashout(days = 90)

attrition(cdm$cohort1) |>
  glimpse()
```

---

summariseDiscontinuationAsSurvival

Summarise discontinuation as a survival analysis

Description

`summariseDiscontinuationAsSurvival()` analyses discontinuation as a survival analysis using the CohortSurvival package. The function assumes that each cohort entry is a continuous treatment era. Discontinuation will be assessed as a survival analysis with index date: *start of the drug treatment era* (`cohort_start_date`) and event of interest: *end of the drug treatment era* (`cohort_end_date`). The analysis will use `estimateSingleEventSurvival()` or `estimateCompetingRiskSurvival()` depending if `competingOutcomeCohortTable` is provided or not.

Usage

```
summariseDiscontinuationAsSurvival(
  cohort,
  cohortId = NULL,
  followUpDays = Inf,
  censorDate = NULL,
  strata = list(),
  competingOutcomeCohortTable = NULL,
  competingOutcomeCohortId = NULL,
  eventGap = 30,
  estimateGap = 1
)
```

Arguments

`cohort`	A cohort_table object.
`cohortId`	A cohort definition id to restrict by. If NULL, all cohorts will be included.
`followUpDays`	Number of days to follow up individuals (lower bound 1, upper bound Inf).
`censorDate`	if not NULL, an individual's follow up will be censored at the given date.

strata — A list of variables to stratify results. These variables must have been added as additional columns in the cohort table.

competingOutcomeCohortTable — The competing outcome cohort table of interest.

competingOutcomeCohortId — Competing outcome cohorts to include. It can either be a cohort_definition_id value or a cohort_name. Multiple ids are allowed.

eventGap — Days between time points for which to report survival events, which are grouped into the specified intervals.

estimateGap — Days between time points for which to report survival estimates. First day will be day zero with risk estimates provided for times up to the end of follow-up, with a gap in days equivalent to eventGap.

Value

A <summarised_result> object that contains the probability to not discontinue over time and the summary statistics. Use tableDiscontinuationAsSurvival() and plotDiscontinuationAsSurvival() to visualise the results.

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

result <- summariseDiscontinuationAsSurvival(cdm$cohort1)

plotDiscontinuationAsSurvival(result)

tableDiscontinuationAsSurvival(result)
```

---

summariseDoseCoverage *Check coverage of daily dose computation in a sample of the cdm for selected concept sets and ingredient*

Description

Check coverage of daily dose computation in a sample of the cdm for selected concept sets and ingredient

Usage

```
summariseDoseCoverage(
  cdm,
  ingredientConceptId,
```

```
  estimates = c("count_missing", "percentage_missing", "mean", "sd", "q25", "median",
    "q75"),
  sampleSize = NULL
)
```

Arguments

cdm	A cdm_reference object.
ingredientConceptId	Ingredient OMOP concept that we are interested for the study.
estimates	Estimates to obtain.
sampleSize	Maximum number of records of an ingredient to estimate dose coverage. If an ingredient has more, a random sample equal to sampleSize will be considered. If NULL, all records will be used.

Value

The function returns information of the coverage of computeDailyDose.R for the selected ingredients and concept sets

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

summariseDoseCoverage(cdm = cdm, ingredientConceptId = 1125315)
```

---

summariseDrugRestart *Summarise the drug restart for each follow-up period of interest.*

Description

Summarise the drug restart for each follow-up period of interest.

Usage

```
summariseDrugRestart(
  cohort,
  cohortId = NULL,
  switchCohortTable,
  switchCohortId = NULL,
  strata = list(),
  followUpDays = Inf,
  censorDate = NULL,
```

```
  incident = TRUE,
  restrictToFirstDiscontinuation = TRUE
)
```

Arguments

cohort	A cohort_table object.
cohortId	A cohort definition id to restrict by. If NULL, all cohorts will be included.
switchCohortTable	A cohort table in the cdm that contains possible alternative treatments.
switchCohortId	The cohort ids to be used from switchCohortTable. If NULL all cohort definition ids are used.
strata	A list of variables to stratify results. These variables must have been added as additional columns in the cohort table.
followUpDays	A vector of number of days to follow up. It can be multiple values.
censorDate	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
incident	Whether the switch treatment has to be incident (start after discontinuation) or not (it can start before the discontinuation and last till after).
restrictToFirstDiscontinuation	Whether to consider only the first discontinuation episode or all of them.

Value

A summarised_result object with the percentages of restart, switch and not exposed per follow-up period given.

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

conceptlist <- list(acetaminophen = 1125360, metformin = c(1503297, 1503327))
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "switch_cohort",
                                        conceptSet = conceptlist)

result <- cdm$cohort1 |>
  summariseDrugRestart(switchCohortTable = "switch_cohort")

tableDrugRestart(result)
```

summariseDrugUtilisation *This function is used to summarise the dose utilisation table over multiple cohorts.*

Description

This function is used to summarise the dose utilisation table over multiple cohorts.

Usage

```
summariseDrugUtilisation(
  cohort,
  cohortId = NULL,
  strata = list(),
  estimates = c("q25", "median", "q75", "mean", "sd", "count_missing",
    "percentage_missing"),
  ingredientConceptId = NULL,
  conceptSet = NULL,
  indexDate = "cohort_start_date",
  censorDate = "cohort_end_date",
  restrictIncident = TRUE,
  gapEra = 1,
  numberExposures = TRUE,
  numberEras = TRUE,
  daysExposed = TRUE,
  daysPrescribed = TRUE,
  timeToExposure = TRUE,
  initialExposureDuration = TRUE,
  initialQuantity = TRUE,
  cumulativeQuantity = TRUE,
  initialDailyDose = TRUE,
  cumulativeDose = TRUE
)
```

Arguments

cohort	A cohort_table object.
cohortId	A cohort definition id to restrict by. If NULL, all cohorts will be included.
strata	A list of variables to stratify results. These variables must have been added as additional columns in the cohort table.
estimates	Estimates that we want for the columns.
ingredientConceptId	Ingredient OMOP concept that we are interested for the study.
conceptSet	List of concepts to be included.

indexDate Name of a column that indicates the date to start the analysis.

censorDate Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.

restrictIncident
Whether to include only incident prescriptions in the analysis. If FALSE all prescriptions that overlap with the study period will be included.

gapEra Number of days between two continuous exposures to be considered in the same era.

numberExposures
Whether to include 'number_exposures' (number of drug exposure records between indexDate and censorDate).

numberEras Whether to include 'number_eras' (number of continuous exposure episodes between indexDate and censorDate).

daysExposed Whether to include 'days_exposed' (number of days that the individual is in a continuous exposure episode, including allowed treatment gaps, between indexDate and censorDate; sum of the length of the different drug eras).

daysPrescribed Whether to include 'days_prescribed' (sum of the number of days for each prescription that contribute in the analysis).

timeToExposure Whether to include 'time_to_exposure' (number of days between indexDate and the first episode).

initialExposureDuration
Whether to include 'initial_exposure_duration' (number of prescribed days of the first drug exposure record).

initialQuantity
Whether to include 'initial_quantity' (quantity of the first drug exposure record).

cumulativeQuantity
Whether to include 'cumulative_quantity' (sum of the quantity of the different exposures considered in the analysis).

initialDailyDose
Whether to include 'initial_daily_dose_{unit}' (daily dose of the first considered prescription).

cumulativeDose Whether to include 'cumulative_dose_{unit}' (sum of the cumulative dose of the analysed drug exposure records).

Value

A summary of drug utilisation stratified by cohort_name and strata_name

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)

cdm <- mockDrugUtilisation()
cdm <- generateIngredientCohortSet(cdm = cdm,
                                   ingredient = "acetaminophen",
```

```
                                    name = "dus_cohort")

cdm$dus_cohort |>
  summariseDrugUtilisation(ingredientConceptId = 1125315)
```

`summariseIndication` *Summarise the indications of individuals in a drug cohort*

Description

Summarise the observed indications of patients in a drug cohort based on their presence in an indication cohort in a specified time window. If an individual is not in one of the indication cohorts, they will be considered to have an unknown indication if they are present in one of the specified OMOP CDM clinical tables. Otherwise, if they are neither in an indication cohort or a clinical table they will be considered as having no observed indication.

Usage

```
summariseIndication(
  cohort,
  strata = list(),
  indicationCohortName,
  cohortId = NULL,
  indicationCohortId = NULL,
  indicationWindow = list(c(0, 0)),
  unknownIndicationTable = NULL,
  indexDate = "cohort_start_date",
  mutuallyExclusive = TRUE,
  censorDate = NULL,
  notInObservation = "include"
)
```

Arguments

`cohort`	A cohort_table object.
`strata`	A list of variables to stratify results. These variables must have been added as additional columns in the cohort table.
`indicationCohortName`	Name of the cohort table with potential indications.
`cohortId`	A cohort definition id to restrict by. If NULL, all cohorts will be included.
`indicationCohortId`	The target cohort ID to add indication. If NULL all cohorts will be considered.
`indicationWindow`	The time window over which to identify indications.

`unknownIndicationTable`
: Tables in the OMOP CDM to search for unknown indications.

`indexDate`
: Name of a column that indicates the date to start the analysis.

`mutuallyExclusive`
: Whether to report indications as mutually exclusive or report them as independent results.

`censorDate`
: Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.

`notInObservation`
: Whether to include the individuals not in observation as a separate category ("include") or to exclude them ("exclude").

Value

A summarised result

Examples

```
library(DrugUtilisation)
library(dplyr, warn.conflicts = FALSE)
library(CDMConnector)

cdm <- mockDrugUtilisation(source = "duckdb")

indications <- list(headache = 378253, asthma = 317009)
cdm <- generateConceptCohortSet(cdm = cdm,
                                conceptSet = indications,
                                name = "indication_cohorts")

cdm <- generateIngredientCohortSet(cdm = cdm,
                                   name = "drug_cohort",
                                   ingredient = "acetaminophen")

cdm$drug_cohort |>
  summariseIndication(
    indicationCohortName = "indication_cohorts",
    unknownIndicationTable = "condition_occurrence",
    indicationWindow = list(c(-Inf, 0))
  ) |>
  glimpse()
```

---

summariseProportionOfPatientsCovered

Summarise proportion Of patients covered

Description

Gives the proportion of patients still in observation who are in the cohort on any given day following their first cohort entry. This is known as the "proportion of patients covered" (PPC) method for assessing treatment persistence.

Usage

```
summariseProportionOfPatientsCovered(
  cohort,
  cohortId = NULL,
  strata = list(),
  followUpDays = NULL
)
```

Arguments

cohort	A cohort_table object.
cohortId	A cohort definition id to restrict by. If NULL, all cohorts will be included.
strata	A list of variables to stratify results. These variables must have been added as additional columns in the cohort table.
followUpDays	Number of days to follow up individuals for. If NULL the maximum amount of days from an individuals first cohort start date to their last cohort end date will be used

Value

A summarised result

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation(numberIndividuals = 100)

result <- cdm$cohort1 |>
  summariseProportionOfPatientsCovered(followUpDays = 365)

tidy(result)
```

`summariseTreatment` *This function is used to summarise treatments received*

Description

This function is used to summarise treatments received

Usage

```
summariseTreatment(
  cohort,
  window,
  treatmentCohortName,
  cohortId = NULL,
  treatmentCohortId = NULL,
  strata = list(),
  indexDate = "cohort_start_date",
  censorDate = NULL,
  mutuallyExclusive = FALSE,
  notInObservation = "include"
)
```

Arguments

`cohort`	A cohort_table object.
`window`	Time window over which to summarise the treatments.
`treatmentCohortName`	Name of a cohort in the cdm that contains the treatments of interest.
`cohortId`	A cohort definition id to restrict by. If NULL, all cohorts will be included.
`treatmentCohortId`	Cohort definition id of interest from treatmentCohortName.
`strata`	A list of variables to stratify results. These variables must have been added as additional columns in the cohort table.
`indexDate`	Name of a column that indicates the date to start the analysis.
`censorDate`	Name of a column that indicates the date to stop the analysis, if NULL end of individuals observation is used.
`mutuallyExclusive`	Whether to include mutually exclusive treatments or not.
`notInObservation`	Whether to include the individuals not in observation as a separate category ("include") or to exclude them ("exclude").

Value

A summary of treatments stratified by cohort_name and strata_name

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()
cdm$cohort1 |>
  summariseTreatment(
    treatmentCohortName = "cohort2",
    window = list(c(0, 30), c(31, 365))
  )
```

`tableDiscontinuationAsSurvival` *Create a table with discontinuation as survival results*

Description

Create a table with discontinuation as survival results

Usage

```
tableDiscontinuationAsSurvival(
  result,
  header = c("cdm_name"),
  groupColumn = c("cohort_name", strataColumns(result)),
  type = NULL,
  gapSummary = TRUE,
  hide = c("variable_level"[!gapSummary], "competing_outcome", "estimate_gap",
    "event_gap", "follow_up_days", "cohort_survival_version"),
  style = NULL,
  .options = list()
)
```

Arguments

`result`	A summarised_result object.
`header`	Columns to use as header. See options with `availableTableColumns(result)`.
`groupColumn`	Columns to group by. See options with `availableTableColumns(result)`.
`type`	Character string specifying the desired output table format. See `visOmopResults::tableType()` for supported table types. If type = `NULL`, global options (set via `visOmopResults::setGlobalTableOpti` will be used if available; otherwise, a default 'gt' table is created.
`gapSummary`	Whether to include *Gap summary* statistics.
`hide`	Columns to hide from the visualisation. See options with `availableTableColumns(result)`.
`style`	Defines the visual formatting of the table. This argument can be provided in one of the following ways:

1. **Pre-defined style**: Use the name of a built-in style (e.g., "darwin"). See `visOmopResults::tableStyle()` for available options.
2. **YAML file path**: Provide the path to an existing .yml file defining a new style.
3. **List of custome R code**: Supply a block of custom R code or a named list describing styles for each table section. This code must be specific to the selected table type.

If style = `NULL`, the function will use global options (see `visOmopResults::setGlobalTableOptions()`) or an existing `_brand.yml` file (if found); otherwise, the default style is applied.
For more details, see the *Styles* vignette in **visOmopResults** website.

`.options` A named list with additional formatting options. `visOmopResults::tableOptions()` shows allowed arguments and their default values.

Value

A table with a formatted version of `summariseDiscontinuationAsSurvival()` results.

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

result <- summariseDiscontinuationAsSurvival(cdm$cohort1)

tableDiscontinuationAsSurvival(result)
```

---

tableDoseCoverage *Format a dose_coverage object into a visual table.*

Description

Format a dose_coverage object into a visual table.

Usage

```
tableDoseCoverage(
  result,
  header = c("variable_name", "estimate_name"),
  groupColumn = c("cdm_name", "ingredient_name"),
  type = NULL,
  hide = c("variable_level", "sample_size"),
  style = NULL,
  .options = list()
)
```

Arguments

result	A summarised_result object.
header	Columns to use as header. See options with `availableTableColumns(result)`.
groupColumn	Columns to group by. See options with `availableTableColumns(result)`.
type	Character string specifying the desired output table format. See `visOmopResults::tableType()` for supported table types. If type = NULL, global options (set via `visOmopResults::setGlobalTableOpti` will be used if available; otherwise, a default 'gt' table is created.
hide	Columns to hide from the visualisation. See options with `availableTableColumns(result)`.
style	Defines the visual formatting of the table. This argument can be provided in one of the following ways: 1. **Pre-defined style**: Use the name of a built-in style (e.g., "darwin"). See `visOmopResults::tableStyle()` for available options. 2. **YAML file path**: Provide the path to an existing .yml file defining a new style. 3. **List of custome R code**: Supply a block of custom R code or a named list describing styles for each table section. This code must be specific to the selected table type. If style = NULL, the function will use global options (see `visOmopResults::setGlobalTableOptions()`) or an existing `_brand.yml` file (if found); otherwise, the default style is applied. For more details, see the *Styles* vignette in **visOmopResults** website.
.options	A named list with additional formatting options. `visOmopResults::tableOptions()` shows allowed arguments and their default values.

Value

A table with a formatted version of summariseDrugCoverage() results.

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

result <- summariseDoseCoverage(cdm, 1125315)

tableDoseCoverage(result)
```

`tableDrugRestart` *Format a drug_restart object into a visual table.*

Description

Format a drug_restart object into a visual table.

Usage

```
tableDrugRestart(
  result,
  header = c("cdm_name", "cohort_name"),
  groupColumn = "variable_name",
  type = NULL,
  hide = c("censor_date", "restrict_to_first_discontinuation", "follow_up_days",
    "cohort_table_name", "incident", "switch_cohort_table"),
  style = NULL,
  .options = list()
)
```

Arguments

`result`	A summarised_result object.
`header`	Columns to use as header. See options with `availableTableColumns(result)`.
`groupColumn`	Columns to group by. See options with `availableTableColumns(result)`.
`type`	Character string specifying the desired output table format. See `visOmopResults::tableType()` for supported table types. If type = `NULL`, global options (set via `visOmopResults::setGlobalTableOpti` will be used if available; otherwise, a default 'gt' table is created.
`hide`	Columns to hide from the visualisation. See options with `availableTableColumns(result)`.
`style`	Defines the visual formatting of the table. This argument can be provided in one of the following ways: 1. **Pre-defined style**: Use the name of a built-in style (e.g., "darwin"). See `visOmopResults::tableStyle()` for available options. 2. **YAML file path**: Provide the path to an existing .yml file defining a new style. 3. **List of custome R code**: Supply a block of custom R code or a named list describing styles for each table section. This code must be specific to the selected table type. If style = `NULL`, the function will use global options (see `visOmopResults::setGlobalTableOptions()`) or an existing `_brand.yml` file (if found); otherwise, the default style is applied. For more details, see the *Styles* vignette in **visOmopResults** website.
`.options`	A named list with additional formatting options. `visOmopResults::tableOptions()` shows allowed arguments and their default values.

Value

A table with a formatted version of summariseDrugRestart() results.

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

conceptlist <- list(acetaminophen = 1125360, metformin = c(1503297, 1503327))
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "switch_cohort",
                                        conceptSet = conceptlist)

result <- cdm$cohort1 |>
  summariseDrugRestart(switchCohortTable = "switch_cohort")

tableDrugRestart(result)
```

---

tableDrugUtilisation *Format a drug_utilisation object into a visual table.*

Description

Format a drug_utilisation object into a visual table.

Usage

```
tableDrugUtilisation(
  result,
  header = c("cdm_name"),
  groupColumn = c("cohort_name", strataColumns(result)),
  type = NULL,
 hide = c("variable_level", "censor_date", "cohort_table_name", "gap_era", "index_date",
    "restrict_incident"),
  style = NULL,
  .options = list()
)
```

Arguments

result	A summarised_result object.
header	Columns to use as header. See options with `availableTableColumns(result)`.
groupColumn	Columns to group by. See options with `availableTableColumns(result)`.

type	Character string specifying the desired output table format. See visOmopResults::tableType() for supported table types. If type = NULL, global options (set via visOmopResults::setGlobalTableOpti will be used if available; otherwise, a default 'gt' table is created.
hide	Columns to hide from the visualisation. See options with availableTableColumns(result).
style	Defines the visual formatting of the table. This argument can be provided in one of the following ways: 1. **Pre-defined style**: Use the name of a built-in style (e.g., "darwin"). See visOmopResults::tableStyle() for available options. 2. **YAML file path**: Provide the path to an existing .yml file defining a new style. 3. **List of custome R code**: Supply a block of custom R code or a named list describing styles for each table section. This code must be specific to the selected table type. If style = NULL, the function will use global options (see visOmopResults::setGlobalTableOptions()) or an existing _brand.yml file (if found); otherwise, the default style is applied. For more details, see the *Styles* vignette in **visOmopResults** website.
.options	A named list with additional formatting options. visOmopResults::tableOptions() shows allowed arguments and their default values.

Value

A table with a formatted version of summariseIndication() results.

Examples

```
library(DrugUtilisation)
library(CodelistGenerator)

cdm <- mockDrugUtilisation()
codelist <- getDrugIngredientCodes(cdm = cdm, name = "acetaminophen")
cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "dus_cohort",
                                        conceptSet = codelist)

drugUse <- cdm$dus_cohort |>
  summariseDrugUtilisation(ingredientConceptId = 1125315)

tableDrugUtilisation(drugUse)
```

---

tableIndication *Create a table showing indication results*

Description

Create a table showing indication results

Usage

```
tableIndication(
  result,
  header = c("cdm_name", "cohort_name", strataColumns(result)),
  groupColumn = "variable_name",
  hide = c("window_name", "mutually_exclusive", "unknown_indication_table",
    "censor_date", "cohort_table_name", "index_date", "indication_cohort_name"),
  type = NULL,
  style = NULL,
  .options = list()
)
```

Arguments

result	A summarised_result object.
header	Columns to use as header. See options with `availableTableColumns(result)`.
groupColumn	Columns to group by. See options with `availableTableColumns(result)`.
hide	Columns to hide from the visualisation. See options with `availableTableColumns(result)`.
type	Character string specifying the desired output table format. See `visOmopResults::tableType()` for supported table types. If type = `NULL`, global options (set via `visOmopResults::setGlobalTableOpti` will be used if available; otherwise, a default 'gt' table is created.
style	Defines the visual formatting of the table. This argument can be provided in one of the following ways: 1. **Pre-defined style**: Use the name of a built-in style (e.g., "darwin"). See `visOmopResults::tableStyle()` for available options. 2. **YAML file path**: Provide the path to an existing .yml file defining a new style. 3. **List of custome R code**: Supply a block of custom R code or a named list describing styles for each table section. This code must be specific to the selected table type. If style = `NULL`, the function will use global options (see `visOmopResults::setGlobalTableOptions()`) or an existing `_brand.yml` file (if found); otherwise, the default style is applied. For more details, see the *Styles* vignette in **visOmopResults** website.
.options	A named list with additional formatting options. `visOmopResults::tableOptions()` shows allowed arguments and their default values.

Value

A table with a formatted version of summariseIndication() results.

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()
```

```
result <- cdm$cohort1 |>
  summariseIndication(
    indicationCohortName = "cohort2",
    indicationWindow = list(c(-30, 0)),
    unknownIndicationTable = "condition_occurrence"
  )

tableIndication(result)
```

`tableProportionOfPatientsCovered` *Create a table with proportion of patients covered results*

Description

Create a table with proportion of patients covered results

Usage

```
tableProportionOfPatientsCovered(
  result,
  header = c("cohort_name", strataColumns(result)),
  groupColumn = "cdm_name",
  type = NULL,
  hide = c("variable_name", "variable_level", "cohort_table_name"),
  style = NULL,
  .options = list()
)
```

Arguments

`result`	A summarised_result object.
`header`	Columns to use as header. See options with `availableTableColumns(result)`.
`groupColumn`	Columns to group by. See options with `availableTableColumns(result)`.
`type`	Character string specifying the desired output table format. See `visOmopResults::tableType()` for supported table types. If type = `NULL`, global options (set via `visOmopResults::setGlobalTableOpti` will be used if available; otherwise, a default 'gt' table is created.
`hide`	Columns to hide from the visualisation. See options with `availableTableColumns(result)`.
`style`	Defines the visual formatting of the table. This argument can be provided in one of the following ways: 1. **Pre-defined style**: Use the name of a built-in style (e.g., "darwin"). See `visOmopResults::tableStyle()` for available options. 2. **YAML file path**: Provide the path to an existing .yml file defining a new style.

3. **List of custome R code**: Supply a block of custom R code or a named list describing styles for each table section. This code must be specific to the selected table type.

If style = NULL, the function will use global options (see `visOmopResults::setGlobalTableOptions()`) or an existing `_brand.yml` file (if found); otherwise, the default style is applied.
For more details, see the *Styles* vignette in **visOmopResults** website.

`.options` A named list with additional formatting options. `visOmopResults::tableOptions()` shows allowed arguments and their default values.

Value

A table with a formatted version of summariseProportionOfPatientsCovered() results.

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

cdm <- generateDrugUtilisationCohortSet(cdm = cdm,
                                        name = "my_cohort",
                                conceptSet = list(drug_of_interest = c(1503297, 1503327)))

result <- cdm$my_cohort |>
  summariseProportionOfPatientsCovered(followUpDays = 365)

tableProportionOfPatientsCovered(result)
```

---

tableTreatment *Format a summarised_treatment result into a visual table.*

Description

Format a summarised_treatment result into a visual table.

Usage

```
tableTreatment(
  result,
  header = c("cdm_name", "cohort_name"),
  groupColumn = "variable_name",
  type = NULL,
 hide = c("window_name", "mutually_exclusive", "censor_date", "cohort_table_name",
    "index_date", "treatment_cohort_name"),
  style = NULL,
  .options = list()
)
```

Arguments

result	A summarised_result object.
header	Columns to use as header. See options with `availableTableColumns(result)`.
groupColumn	Columns to group by. See options with `availableTableColumns(result)`.
type	Character string specifying the desired output table format. See `visOmopResults::tableType()` for supported table types. If type = NULL, global options (set via `visOmopResults::setGlobalTableOpti` will be used if available; otherwise, a default 'gt' table is created.
hide	Columns to hide from the visualisation. See options with `availableTableColumns(result)`.
style	Defines the visual formatting of the table. This argument can be provided in one of the following ways: 1. **Pre-defined style**: Use the name of a built-in style (e.g., "darwin"). See `visOmopResults::tableStyle()` for available options. 2. **YAML file path**: Provide the path to an existing .yml file defining a new style. 3. **List of custome R code**: Supply a block of custom R code or a named list describing styles for each table section. This code must be specific to the selected table type. If style = NULL, the function will use global options (see `visOmopResults::setGlobalTableOptions()`) or an existing `_brand.yml` file (if found); otherwise, the default style is applied. For more details, see the *Styles* vignette in **visOmopResults** website.
.options	A named list with additional formatting options. `visOmopResults::tableOptions()` shows allowed arguments and their default values.

Value

A table with a formatted version of summariseTreatment() results.

Examples

```
library(DrugUtilisation)

cdm <- mockDrugUtilisation()

result <- cdm$cohort1 |>
  summariseTreatment(
    treatmentCohortName = "cohort2",
    window = list(c(0, 30), c(31, 365))
  )

tableTreatment(result)
```

Index