Package 'CohortConstructor'

July 21, 2025

Title Build and Manipulate Study Cohorts Using a Common Data Model

Version 0.4.0

Description Create and manipulate study cohorts in data mapped to the Observational Medical Outcomes Partnership Common Data Model.

License Apache License (>= 2)

Encoding UTF-8

RoxygenNote 7.3.2

Imports CDMConnector (>= 1.7.0), checkmate, cli, clock, dbplyr (>= 2.5.0), dplyr, glue, magrittr, omopgenerics (>= 1.0.0), PatientProfiles (>= 1.2.3), purrr, rlang, tidyr, utils

Suggests DBI, CodelistGenerator (>= 3.4.1), DrugUtilisation, duckdb, knitr, rmarkdown, testthat (>= 3.0.0), tibble, stringr, IncidencePrevalence, omock (>= 0.2.0), covr, RPostgres, odbc, CohortCharacteristics, ggplot2, DiagrammeR, visOmopResults, gt, scales, here, ggpubr, SqlRender, CirceR, tictoc

Config/testthat/edition 3

Config/testthat/parallel true

VignetteBuilder knitr

Depends R (>= 4.1)

URL https://ohdsi.github.io/CohortConstructor/, https://github.com/OHDSI/CohortConstructor

LazyData true

NeedsCompilation no

Author Edward Burn [aut, cre] (ORCID: <https://orcid.org/0000-0002-9286-1128>), Marti Catala [aut] (ORCID: <https://orcid.org/0000-0003-3308-9905>), Nuria Mercade-Besora [aut] (ORCID: <https://orcid.org/0009-0006-7948-3747>), Marta Alcalde-Herraiz [aut] (ORCID: <https://orcid.org/0009-0002-4405-1814>), Mike Du [aut] (ORCID: <https://orcid.org/0000-0002-9517-8834>), Yuchen Guo [aut] (ORCID: <https://orcid.org/0000-0002-0847-4855>),

Xihang Chen [aut] (ORCID: <https://orcid.org/0009-0001-8112-8959>),
Kim Lopez-Guell [aut] (ORCID: <https://orcid.org/0000-0002-8462-8668>),
Elin Rowlands [aut] (ORCID: <https://orcid.org/0009-0005-5166-0417>)

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

Repository CRAN

Date/Publication 2025-05-08 10:00:02 UTC

Contents

`addCohortTableIndex` *Add an index to a cohort table*

Description

Adds an index on subject_id and cohort_start_date to a cohort table. Note, currently only indexes will be added if the table is in a postgres database.

Usage

```
addCohortTableIndex(cohort)
```

Arguments

`cohort`	A cohort table in a cdm reference.

Value

The cohort table

`benchmarkCohortConstructor`
Run benchmark of CohortConstructor package

Description

Run benchmark of CohortConstructor cohort instantiation time compared to CIRCE from JSON. More information in the benchmarking vignette.

Usage

```
benchmarkCohortConstructor(
  cdm,
  runCIRCE = TRUE,
  runCohortConstructorDefinition = TRUE,
  runCohortConstructorDomain = TRUE,
  dropCohorts = TRUE
)
```

Arguments

cdm	A cdm reference.
runCIRCE	Whether to run cohorts from JSON definitions generated with Atlas.
runCohortConstructorDefinition	Whether to run the benchmark part where cohorts are created with CohortConstructor by definition (one by one, separately).
runCohortConstructorDomain	Whether to run the benchmark part where cohorts are created with CohortConstructor by domain (instantianting base cohort all together, as a set).
dropCohorts	Whether to drop cohorts created during benchmark.

---

benchmarkData *Benchmarking results*

Description

Benchmarking results

Usage

```
benchmarkData
```

Format

A list of results from benchmarking

---

collapseCohorts *Collapse cohort entries using a certain gap to concatenate records.*

Description

`collapseCohorts()` concatenates cohort records, allowing for some number of days between one finishing and the next starting.

Usage

```
collapseCohorts(
  cohort,
  cohortId = NULL,
  gap = 0,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

cohort	A cohort table in a cdm reference.
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
gap	Number of days between two subsequent cohort entries to be merged in a single cohort record.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

A cohort table

---

conceptCohort *Create cohorts based on a concept set*

Description

`conceptCohort()` creates a cohort table from patient records from the clinical tables in the OMOP CDM.

The following tables are currently supported for creating concept cohorts:

- condition_occurrence
- device_exposure
- drug_exposure
- measurement
- observation
- procedure_occurrence
- visit_occurrence

Cohort duration is based on record start and end (e.g. condition_start_date and condition_end_date for records coming from the condition_occurrence tables). So that the resulting table satisfies the requirements of an OMOP CDM cohort table:

- Cohort entries will not overlap. Overlapping records will be combined based on the overlap argument.
- Cohort entries will not go out of observation. If a record starts outside of an observation period it will be silently ignored. If a record ends outside of an observation period it will be trimmed so as to end at the preceding observation period end date.

Usage

```
conceptCohort(
  cdm,
  conceptSet,
  name,
  exit = "event_end_date",
  overlap = "merge",
  inObservation = TRUE,
  table = NULL,
  useSourceFields = FALSE,
  subsetCohort = NULL,
  subsetCohortId = NULL
)
```

Arguments

cdm	A cdm reference.
conceptSet	A conceptSet, which can either be a codelist or a conceptSetExpression.
name	Name of the new cohort table created in the cdm object.
exit	How the cohort end date is defined. Can be either "event_end_date" or "event_start_date".
overlap	How to deal with overlapping records. In all cases cohort start will be set as the earliest start date. If "merge", cohort end will be the latest end date. If "extend", cohort end date will be set by adding together the total days from each of the overlapping records.
inObservation	If TRUE, only records in observation will be used. If FALSE, records before the start of observation period will be considered, with startdate the start of observation.
table	Name of OMOP tables to search for records of the concepts provided. If NULL, each concept will be search at the assigned domain in the concept table.
useSourceFields	If TRUE, the source concept_id fields will also be used when identifying relevant clinical records. If FALSE, only the standard concept_id fields will be used.
subsetCohort	A character refering to a cohort table containing individuals for whom cohorts will be generated. Only individuals in this table will appear in the generated cohort.
subsetCohortId	Optional. Specifies cohort IDs from the subsetCohort table to include. If none are provided, all cohorts from the subsetCohort are included.

Value

A cohort table

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor(conditionOccurrence = TRUE, drugExposure = TRUE)

cdm$cohort <- conceptCohort(cdm = cdm, conceptSet = list(a = 444074), name = "cohort")

cdm$cohort |> attrition()

# Create a cohort based on a concept set. The cohort exit is set to the event start date.
# If two records overlap, the cohort end date is set as the sum of the duration of
# all overlapping records. Only individuals included in the existing `cohort` will be considered.

conceptSet <- list("nitrogen" = c(35604434, 35604439),
"potassium" = c(40741270, 42899580, 44081436))

cohort_drugs <- conceptCohort(cdm,
                              conceptSet = conceptSet,
                              name = "cohort_drugs",
                              exit = "event_start_date",
                              overlap = "extend",
                              subsetCohort = "cohort"
)

cohort_drugs |> attrition()
```

`copyCohorts` *Copy a cohort table*

Description

`copyCohorts()` copies an existing cohort table to a new location.

Usage

```
copyCohorts(cohort, name, n = 1, cohortId = NULL, .softValidation = TRUE)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`name`	Name of the new cohort table created in the cdm object.
`n`	Number of times to duplicate the selected cohorts.
`cohortId`	Vector identifying which cohorts to include (cohort_definition_id or cohort_name). Cohorts not included will be removed from the cohort set.

`.softValidation`
: Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

A new cohort table containing cohorts from the original cohort table.

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
cdm$cohort3 <- copyCohorts(cdm$cohort1, n = 2, cohortId = 1, name = "cohort3")
```

---

deathCohort *Create cohort based on the death table*

Description

Create cohort based on the death table

Usage

```
deathCohort(cdm, name, subsetCohort = NULL, subsetCohortId = NULL)
```

Arguments

`cdm`	A cdm reference.
`name`	Name of the new cohort table created in the cdm object.
`subsetCohort`	A character refering to a cohort table containing individuals for whom cohorts will be generated. Only individuals in this table will appear in the generated cohort.
`subsetCohortId`	Optional. Specifies cohort IDs from the `subsetCohort` table to include. If none are provided, all cohorts from the `subsetCohort` are included.

Value

A cohort table with a death cohort in cdm

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor(death = TRUE)

# Generate a death cohort
death_cohort <- deathCohort(cdm, name = "death_cohort")
death_cohort

# Create a death cohort for females aged over 50 years old.

# Create a demographics cohort with age range and sex filters
cdm$my_cohort <- demographicsCohort(cdm, "my_cohort", ageRange = c(50,100), sex = "Female")

# Generate a death cohort, restricted to individuals in 'my_cohort'
death_cohort <- deathCohort(cdm, name = "death_cohort", subsetCohort = "my_cohort")
death_cohort |> attrition()
```

`demographicsCohort` *Create cohorts based on patient demographics*

Description

`demographicsCohort()` creates a cohort table based on patient characteristics. If and when an individual satisfies all the criteria they enter the cohort. When they stop satisfying any of the criteria their cohort entry ends.

Usage

```
demographicsCohort(
  cdm,
  name,
  ageRange = NULL,
  sex = NULL,
  minPriorObservation = NULL,
  .softValidation = TRUE
)
```

Arguments

`cdm`	A cdm reference.
`name`	Name of the new cohort table created in the cdm object.
`ageRange`	A list of vectors specifying minimum and maximum age.
`sex`	Can be "Both", "Male" or "Female".

`minPriorObservation`
: A minimum number of continuous prior observation days in the database.

`.softValidation`
: Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

A cohort table

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor()

cohort <-  cdm |>
    demographicsCohort(name = "cohort3", ageRange = c(18,40), sex = "Male")

attrition(cohort)

# Can also create multiple demographic cohorts, and add minimum prior history requirements.

cohort <- cdm |>
    demographicsCohort(name = "cohort4",
    ageRange = list(c(0, 19),c(20, 64),c(65, 150)),
    sex = c("Male", "Female", "Both"),
    minPriorObservation = 365)

attrition(cohort)
```

---

entryAtFirstDate *Update cohort start date to be the first date from of a set of column dates*

Description

`entryAtFirstDate()` resets cohort start date based on a set of specified column dates. The first date that occurs is chosen.

Usage

```
entryAtFirstDate(
  cohort,
  dateColumns,
```

```
  cohortId = NULL,
  returnReason = TRUE,
  keepDateColumns = TRUE,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

cohort	A cohort table in a cdm reference.
dateColumns	Character vector indicating date columns in the cohort table to consider.
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
returnReason	If TRUE it will return a column indicating which of the dateColumns was used.
keepDateColumns	If TRUE the returned cohort will keep columns in dateColumns.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table.

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor(tables = list(
"cohort" = dplyr::tibble(
  cohort_definition_id = 1,
  subject_id = c(1, 2, 3, 4),
  cohort_start_date = as.Date(c("2000-06-03", "2000-01-01", "2015-01-15", "2000-12-09")),
  cohort_end_date = as.Date(c("2001-09-01", "2001-01-12", "2015-02-15", "2002-12-09")),
  date_1 = as.Date(c("2001-08-01", "2001-01-01", "2015-01-15", "2002-12-09")),
  date_2 = as.Date(c("2001-08-01", NA, "2015-02-14", "2002-12-09"))
)
))
cdm$cohort |> entryAtLastDate(dateColumns = c("date_1", "date_2"))
```

`entryAtLastDate` *Set cohort start date to the last of a set of column dates*

Description

`entryAtLastDate()` resets cohort end date based on a set of specified column dates. The last date is chosen.

Usage

```
entryAtLastDate(
  cohort,
  dateColumns,
  cohortId = NULL,
  returnReason = TRUE,
  keepDateColumns = TRUE,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`dateColumns`	Character vector indicating date columns in the cohort table to consider.
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`returnReason`	If TRUE it will return a column indicating which of the `dateColumns` was used.
`keepDateColumns`	If TRUE the returned cohort will keep columns in `dateColumns`.
`name`	Name of the new cohort table created in the cdm object.
`.softValidation`	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table.

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor(tables = list(
"cohort" = dplyr::tibble(
  cohort_definition_id = 1,
  subject_id = c(1, 2, 3, 4),
  cohort_start_date = as.Date(c("2000-06-03", "2000-01-01", "2015-01-15", "2000-12-09")),
  cohort_end_date = as.Date(c("2001-09-01", "2001-01-12", "2015-02-15", "2002-12-09")),
  date_1 = as.Date(c("2001-08-01", "2001-01-01", "2015-01-15", "2002-12-09")),
  date_2 = as.Date(c("2001-08-01", NA, "2015-02-14", "2002-12-09"))
)
))
cdm$cohort |> entryAtLastDate(dateColumns = c("date_1", "date_2"))
```

---

exitAtDeath *Set cohort end date to death date*

Description

This functions changes cohort end date to subject's death date. In the case were this generates overlapping records in the cohort, those overlapping entries will be merged.

Usage

```
exitAtDeath(
  cohort,
  cohortId = NULL,
  requireDeath = FALSE,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

cohort	A cohort table in a cdm reference.
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
requireDeath	If TRUE, subjects without a death record will be dropped, while if FALSE their end date will be left as is.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table.

Examples

```
library(PatientProfiles)
library(CohortConstructor)
cdm <- mockPatientProfiles()
cdm$cohort1 |> exitAtDeath()
```

`exitAtFirstDate` *Set cohort end date to the first of a set of column dates*

Description

`exitAtFirstDate()` resets cohort end date based on a set of specified column dates. The first date that occurs is chosen.

Usage

```
exitAtFirstDate(
  cohort,
  dateColumns,
  cohortId = NULL,
  returnReason = TRUE,
  keepDateColumns = TRUE,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`dateColumns`	Character vector indicating date columns in the cohort table to consider.
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`returnReason`	If TRUE it will return a column indicating which of the `dateColumns` was used.
`keepDateColumns`	If TRUE the returned cohort will keep columns in `dateColumns`.
`name`	Name of the new cohort table created in the cdm object.
`.softValidation`	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table.

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor(tables = list(
"cohort" = dplyr::tibble(
  cohort_definition_id = 1,
  subject_id = c(1, 2, 3, 4),
  cohort_start_date = as.Date(c("2000-06-03", "2000-01-01", "2015-01-15", "2000-12-09")),
  cohort_end_date = as.Date(c("2001-09-01", "2001-01-12", "2015-02-15", "2002-12-09")),
  date_1 = as.Date(c("2001-08-01", "2001-01-01", "2015-01-15", "2002-12-09")),
  date_2 = as.Date(c("2001-08-01", NA, "2015-04-15", "2002-12-09"))
)
))
cdm$cohort |> exitAtFirstDate(dateColumns = c("date_1", "date_2"))
```

---

exitAtLastDate *Set cohort end date to the last of a set of column dates*

Description

`exitAtLastDate()` resets cohort end date based on a set of specified column dates. The last date that occurs is chosen.

Usage

```
exitAtLastDate(
  cohort,
  dateColumns,
  cohortId = NULL,
  returnReason = TRUE,
  keepDateColumns = TRUE,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`dateColumns`	Character vector indicating date columns in the cohort table to consider.
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`returnReason`	If TRUE it will return a column indicating which of the `dateColumns` was used.

keepDateColumns
: If TRUE the returned cohort will keep columns in dateColumns.

name
: Name of the new cohort table created in the cdm object.

.softValidation
: Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table.

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor(tables = list(
"cohort" = dplyr::tibble(
  cohort_definition_id = 1,
  subject_id = c(1, 2, 3, 4),
  cohort_start_date = as.Date(c("2000-06-03", "2000-01-01", "2015-01-15", "2000-12-09")),
  cohort_end_date = as.Date(c("2001-09-01", "2001-01-12", "2015-02-15", "2002-12-09")),
  date_1 = as.Date(c("2001-08-01", "2001-01-01", "2015-01-15", "2002-12-09")),
  date_2 = as.Date(c("2001-08-01", NA, "2015-04-15", "2002-12-09"))
)
))
cdm$cohort |> exitAtLastDate(dateColumns = c("date_1", "date_2"))
```

---

exitAtObservationEnd *Set cohort end date to end of observation*

Description

exitAtObservationEnd() resets cohort end date based on a set of specified column dates. The last date that occurs is chosen.

This functions changes cohort end date to the end date of the observation period corresponding to the cohort entry. In the case were this generates overlapping records in the cohort, overlapping entries will be merged.

Usage

```
exitAtObservationEnd(
  cohort,
  cohortId = NULL,
  limitToCurrentPeriod = TRUE,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

cohort	A cohort table in a cdm reference.
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
limitToCurrentPeriod	If TRUE, limits the cohort to one entry per person, ending at the current observation period. If FALSE, subsequent observation periods will create new cohort entries.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table.

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor()
cdm$cohort1 |> exitAtObservationEnd()
```

---

intersectCohorts *Generate a combination cohort set between the intersection of different cohorts.*

Description

`intersectCohorts()` combines different cohort entries, with those records that overlap combined and kept. Cohort entries are when an individual was in *both* of the cohorts.

Usage

```
intersectCohorts(
  cohort,
  cohortId = NULL,
  gap = 0,
  returnNonOverlappingCohorts = FALSE,
  keepOriginalCohorts = FALSE,
```

```
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

cohort	A cohort table in a cdm reference.
cohortId	Vector identifying which cohorts to include (cohort_definition_id or cohort_name). Cohorts not included will be removed from the cohort set.
gap	Number of days between two subsequent cohort entries to be merged in a single cohort record.
returnNonOverlappingCohorts	Whether the generated cohorts are mutually exclusive or not.
keepOriginalCohorts	If TRUE the original cohorts will be return together with the new ones. If FALSE only the new cohort will be returned.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

A cohort table.

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor(nPerson = 100)

cdm$cohort3 <- intersectCohorts(
  cohort = cdm$cohort2,
  name = "cohort3",
)

settings(cdm$cohort3)
```

`matchCohorts` *Generate a new cohort matched cohort*

Description

`matchCohorts()` generate a new cohort matched to individuals in an existing cohort. Individuals can be matched based on year of birth and sex. Matching is done at the record level, so if individuals have multiple cohort entries they can be matched to different individuals for each of their records.

Two new cohorts will be created when matching. The first is those cohort entries which were matched ("_sampled" is added to the original cohort name for this cohort). The other is the matches found from the database population ("_matched" is added to the original cohort name for this cohort).

Usage

```
matchCohorts(
  cohort,
  cohortId = NULL,
  matchSex = TRUE,
  matchYearOfBirth = TRUE,
  ratio = 1,
  keepOriginalCohorts = FALSE,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`cohortId`	Vector identifying which cohorts to include (cohort_definition_id or cohort_name). Cohorts not included will be removed from the cohort set.
`matchSex`	Whether to match in sex.
`matchYearOfBirth`	Whether to match in year of birth.
`ratio`	Number of allowed matches per individual in the target cohort.
`keepOriginalCohorts`	If TRUE the original cohorts will be return together with the new ones. If FALSE only the new cohort will be returned.
`name`	Name of the new cohort table created in the cdm object.
`.softValidation`	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

A cohort table.

Examples

```
library(CohortConstructor)
library(dplyr)
cdm <- mockCohortConstructor(nPerson = 200)
cdm$new_matched_cohort <- cdm$cohort2 |>
  matchCohorts(
    name = "new_matched_cohort",
    cohortId = 2,
    matchSex = TRUE,
    matchYearOfBirth = TRUE,
    ratio = 1)
cdm$new_matched_cohort
```

`measurementCohort` *Create measurement-based cohorts*

Description

`measurementCohort()` creates cohorts based on patient records contained in the measurement table. This function extends the `conceptCohort()` as it allows for measurement values associated with the records to be specified.

- If `valueAsConcept` and `valueAsNumber` are NULL then no requirements on of the values associated with measurement records and using `measurementCohort()` will lead to the same result as using `conceptCohort()` (so long as all concepts are from the measurement domain).
- If one of `valueAsConcept` and `valueAsNumber` is not NULL then records will be required to have values that satisfy the requirement specified.
- If both `valueAsConcept` and `valueAsNumber` are not NULL, records will be required to have values that fulfill *either* of the requirements

Usage

```
measurementCohort(
  cdm,
  conceptSet,
  name,
  valueAsConcept = NULL,
  valueAsNumber = NULL,
  table = c("measurement", "observation"),
  inObservation = TRUE
)
```

Arguments

cdm	A cdm reference.
conceptSet	A conceptSet, which can either be a codelist or a conceptSetExpression.
name	Name of the new cohort table created in the cdm object.
valueAsConcept	A vector of cohort IDs used to filter measurements. Only measurements with these values in the value_as_concept_id column of the measurement table will be included. If NULL all entries independent of their value as concept will be considered.
valueAsNumber	A list indicating the range of values and the unit they correspond to, as follows: list("unit_concept_id" = c(rangeValue1, rangeValue2)). If no name is supplied in the list, no requirement on unit concept id will be applied. If NULL, all entries independent of their value as number will be included.
table	Name of OMOP tables to search for records of the concepts provided. Options are "measurement" and/or "observation".
inObservation	If TRUE, only records in observation will be used. If FALSE, records before the start of observation period will be considered, with startdate the start of observation.

Value

A cohort table

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor(con = NULL)
cdm$concept <- cdm$concept |>
  dplyr::union_all(
    dplyr::tibble(
      concept_id = c(4326744, 4298393, 45770407, 8876, 4124457),
      concept_name = c("Blood pressure", "Systemic blood pressure",
                       "Baseline blood pressure", "millimeter mercury column",
                       "Normal range"),
      domain_id = "Measurement",
      vocabulary_id = c("SNOMED", "SNOMED", "SNOMED", "UCUM", "SNOMED"),
      standard_concept = "S",
      concept_class_id = c("Observable Entity", "Observable Entity",
                           "Observable Entity", "Unit", "Qualifier Value"),
      concept_code = NA,
      valid_start_date = NA,
      valid_end_date = NA,
      invalid_reason = NA
    )
  )
cdm$measurement <- dplyr::tibble(
  measurement_id = 1:4,
  person_id = c(1, 1, 2, 3),
  measurement_concept_id = c(4326744, 4298393, 4298393, 45770407),
  measurement_date = as.Date(c("2000-07-01", "2000-12-11", "2002-09-08",
```

```
  "2015-02-19")),
  measurement_type_concept_id = NA,
  value_as_number = c(100, 125, NA, NA),
  value_as_concept_id = c(0, 0, 0, 4124457),
  unit_concept_id = c(8876, 8876, 0, 0)
)
cdm <- CDMConnector::copyCdmTo(
  con = DBI::dbConnect(duckdb::duckdb()),
  cdm = cdm, schema = "main")

cdm$cohort <- measurementCohort(
  cdm = cdm,
  name = "cohort",
  conceptSet = list("normal_blood_pressure" = c(4326744, 4298393, 45770407)),
  valueAsConcept = c(4124457),
  valueAsNumber = list("8876" = c(70, 120)),
  inObservation = TRUE
)

cdm$cohort

# You can also create multiple measurement cohorts, and include records
# outside the observation period.

cdm$cohort2 <- measurementCohort(
  cdm = cdm,
  name = "cohort2",
  conceptSet = list("normal_blood_pressure" = c(4326744, 4298393, 45770407),
                    "high_blood_pressure" = c(4326744, 4298393, 45770407)),
  valueAsConcept = c(4124457),
  valueAsNumber = list("8876" = c(70, 120),
                       "8876" = c(121, 200)),
  inObservation = FALSE
)

cdm$cohort2
```

`mockCohortConstructor` *Function to create a mock cdm reference for CohortConstructor*

Description

`mockCohortConstructor()` creates an example dataset that can be used for demonstrating and testing the package

Usage

```
mockCohortConstructor(
```

```
  nPerson = 10,
  conceptTable = NULL,
  tables = NULL,
  conceptId = NULL,
  conceptIdClass = NULL,
  drugExposure = FALSE,
  conditionOccurrence = FALSE,
  measurement = FALSE,
  death = FALSE,
  otherTables = NULL,
  con = DBI::dbConnect(duckdb::duckdb()),
  writeSchema = "main",
  seed = 123
)
```

Arguments

nPerson	number of person in the cdm
conceptTable	user defined concept table
tables	list of tables to include in the cdm
conceptId	list of concept id
conceptIdClass	the domain class of the conceptId
drugExposure	T/F include drug exposure table in the cdm
conditionOccurrence	T/F include condition occurrence in the cdm
measurement	T/F include measurement in the cdm
death	T/F include death table in the cdm
otherTables	it takes a list of single tibble with names to include other tables in the cdm
con	A DBI connection to create the cdm mock object.
writeSchema	Name of an schema on the same connection with writing permissions.
seed	Seed passed to omock::mockCdmFromTable

Value

cdm object

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor()

cdm
```

padCohortDate *Set cohort start or cohort end*

Description

Set cohort start or cohort end

Usage

```
padCohortDate(
  cohort,
  days,
  cohortDate = "cohort_start_date",
  indexDate = "cohort_start_date",
  collapse = TRUE,
  padObservation = TRUE,
  cohortId = NULL,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

cohort	A cohort table in a cdm reference.
days	Integer with the number of days to add or name of a column (that must be numeric) to add.
cohortDate	'cohort_start_date' or 'cohort_end_date'.
indexDate	Variable in cohort that contains the index date to add.
collapse	Whether to collapse the overlapping records (TRUE) or drop the records that have an ongoing prior record.
padObservation	Whether to pad observations if they are outside observation_period (TRUE) or drop the records if they are outside observation_period (FALSE)
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

Cohort table

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
cdm$cohort1 |>
  padCohortDate(
    cohortDate = "cohort_end_date",
    indexDate = "cohort_start_date",
    days = 10)
```

---

padCohortEnd *Add days to cohort end*

Description

`padCohortEnd()` Adds (or subtracts) a certain number of days to the cohort end date. Note:

- If the days added means that cohort end would be after observation period end date, then observation period end date will be used for cohort exit.
- If the days added means that cohort exit would be after the next cohort start then these overlapping cohort entries will be collapsed.
- If days subtracted means that cohort end would be before cohort start then the cohort entry will be dropped.

Usage

```
padCohortEnd(
  cohort,
  days,
  collapse = TRUE,
  padObservation = TRUE,
  cohortId = NULL,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`days`	Integer with the number of days to add or name of a column (that must be numeric) to add.
`collapse`	Whether to collapse the overlapping records (TRUE) or drop the records that have an ongoing prior record.
`padObservation`	Whether to pad observations if they are outside observation_period (TRUE) or drop the records if they are outside observation_period (FALSE)

cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

Cohort table

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
# add 10 days to each cohort exit
cdm$cohort1 |>
  padCohortEnd(days = 10)
```

---

padCohortStart *Add days to cohort start*

Description

`padCohortStart()` Adds (or subtracts) a certain number of days to the cohort start date. Note:

- If the days added means that cohort start would be after cohort end then the cohort entry will be dropped.
- If subtracting day means that cohort start would be before observation period start then the cohort entry will be dropped.

Usage

```
padCohortStart(
  cohort,
  days,
  collapse = TRUE,
  padObservation = TRUE,
  cohortId = NULL,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

cohort	A cohort table in a cdm reference.
days	Integer with the number of days to add or name of a column (that must be numeric) to add.
collapse	Whether to collapse the overlapping records (TRUE) or drop the records that have an ongoing prior record.
padObservation	Whether to pad observations if they are outside observation_period (TRUE) or drop the records if they are outside observation_period (FALSE)
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

Cohort table

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
# add 10 days to each cohort entry
cdm$cohort1 |>
  padCohortStart(days = 10)
```

---

renameCohort *Utility function to change the name of a cohort.*

Description

Utility function to change the name of a cohort.

Usage

```
renameCohort(cohort, cohortId, newCohortName, .softValidation = TRUE)
```

Arguments

cohort	A cohort table in a cdm reference.
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
newCohortName	Character vector with same
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

A cohort_table object.

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor(nPerson = 100)

settings(cdm$cohort1)

cdm$cohort1 <- cdm$cohort1 |>
  renameCohort(cohortId = 1, newCohortName = "new_name")

settings(cdm$cohort1)
```

---

requireAge *Restrict cohort on age*

Description

`requireAge()` filters cohort records, keeping only records where individuals satisfy the specified age criteria.

Usage

```
requireAge(
  cohort,
  ageRange,
  cohortId = NULL,
  indexDate = "cohort_start_date",
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

cohort	A cohort table in a cdm reference.
ageRange	A list of vectors specifying minimum and maximum age.
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
indexDate	Variable in cohort that contains the date to compute the demographics characteristics on which to restrict on.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table with only records for individuals satisfying the age requirement

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
cdm$cohort1 |>
  requireAge(indexDate = "cohort_start_date",
             ageRange = list(c(18, 65)))
```

---

requireCohortIntersect

Require cohort subjects are present (or absence) in another cohort

---

Description

`requireCohortIntersect()` filters a cohort table based on a requirement that an individual is seen (or not seen) in another cohort in some time window around an index date.

Usage

```
requireCohortIntersect(
  cohort,
  targetCohortTable,
  window,
  intersections = c(1, Inf),
  cohortId = NULL,
```

```
  targetCohortId = NULL,
  indexDate = "cohort_start_date",
  targetStartDate = "cohort_start_date",
  targetEndDate = "cohort_end_date",
  censorDate = NULL,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

cohort	A cohort table in a cdm reference.
targetCohortTable	Name of the cohort that we want to check for intersect.
window	A list of vectors specifying minimum and maximum days from indexDate to consider events over.
intersections	A range indicating number of intersections for criteria to be fulfilled. If a single number is passed, the number of intersections must match this.
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
targetCohortId	Vector of cohort definition ids to include.
indexDate	Name of the column in the cohort that contains the date to compute the intersection.
targetStartDate	Start date of reference in cohort table.
targetEndDate	End date of reference in cohort table. If NULL, incidence of target event in the window will be considered as intersection, otherwise prevalence of that event will be used as intersection (overlap between cohort and event).
censorDate	Whether to censor overlap events at a specific date or a column date of the cohort.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

Cohort table with only those entries satisfying the criteria

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
cdm$cohort1 |>
  requireCohortIntersect(targetCohortTable = "cohort2",
                         targetCohortId = 1,
                         indexDate = "cohort_start_date",
                         window = c(-Inf, 0))
```

`requireConceptIntersect` *Require cohort subjects to have (or not have) events of a concept list*

Description

`requireConceptIntersect()` filters a cohort table based on a requirement that an individual is seen (or not seen) to have events related to a concept list in some time window around an index date.

Usage

```
requireConceptIntersect(
  cohort,
  conceptSet,
  window,
  intersections = c(1, Inf),
  cohortId = NULL,
  indexDate = "cohort_start_date",
  targetStartDate = "event_start_date",
  targetEndDate = "event_end_date",
  inObservation = TRUE,
  censorDate = NULL,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`conceptSet`	A conceptSet, which can either be a codelist or a conceptSetExpression.
`window`	A list of vectors specifying minimum and maximum days from `indexDate` to consider events over.
`intersections`	A range indicating number of intersections for criteria to be fulfilled. If a single number is passed, the number of intersections must match this.

cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
indexDate	Name of the column in the cohort that contains the date to compute the intersection.
targetStartDate	Start date of reference in cohort table.
targetEndDate	End date of reference in cohort table. If NULL, incidence of target event in the window will be considered as intersection, otherwise prevalence of that event will be used as intersection (overlap between cohort and event).
inObservation	If TRUE only records inside an observation period will be considered.
censorDate	Whether to censor overlap events at a specific date or a column date of the cohort.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

Cohort table with only those with the events in the concept list kept (or those without the event if negate = TRUE)

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor(conditionOccurrence = TRUE)
cdm$cohort2 <-  requireConceptIntersect(
  cohort = cdm$cohort1,
  conceptSet = list(a = 194152),
  window = c(-Inf, 0),
  name = "cohort2")
```

---

requireDemographics *Restrict cohort on patient demographics*

Description

`requireDemographics()` filters cohort records, keeping only records where individuals satisfy the specified demographic criteria.

Usage

```
requireDemographics(
  cohort,
  cohortId = NULL,
  indexDate = "cohort_start_date",
  ageRange = list(c(0, 150)),
  sex = c("Both"),
  minPriorObservation = 0,
  minFutureObservation = 0,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

cohort	A cohort table in a cdm reference.
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
indexDate	Variable in cohort that contains the date to compute the demographics characteristics on which to restrict on.
ageRange	A list of vectors specifying minimum and maximum age.
sex	Can be "Both", "Male" or "Female".
minPriorObservation	A minimum number of continuous prior observation days in the database.
minFutureObservation	A minimum number of continuous future observation days in the database.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table with only records for individuals satisfying the demographic requirements

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor(nPerson = 100)
cdm$cohort1 |>
  requireDemographics(indexDate = "cohort_start_date",
                      ageRange = list(c(18, 65)),
                      sex = "Female",
```

```
                        minPriorObservation = 365)
```

`requireFutureObservation` *Restrict cohort on future observation*

Description

`requireFutureObservation()` filters cohort records, keeping only records where individuals satisfy the specified future observation criteria.

Usage

```
requireFutureObservation(
  cohort,
  minFutureObservation,
  cohortId = NULL,
  indexDate = "cohort_start_date",
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`minFutureObservation`	A minimum number of continuous future observation days in the database.
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`indexDate`	Variable in cohort that contains the date to compute the demographics characteristics on which to restrict on.
`name`	Name of the new cohort table created in the cdm object.
`.softValidation`	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table with only records for individuals satisfying the future observation requirement

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
cdm$cohort1 |>
  requireFutureObservation(indexDate = "cohort_start_date",
                           minFutureObservation = 30)
```

`requireInDateRange` *Require that an index date is within a date range*

Description

`requireInDateRange()` filters cohort records, keeping only those for which the index date is within the specified date range.

Usage

```
requireInDateRange(
  cohort,
  dateRange,
  cohortId = NULL,
  indexDate = "cohort_start_date",
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`dateRange`	A date vector with the minimum and maximum dates between which the index date must have been observed.
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`indexDate`	Name of the column in the cohort that contains the date of interest.
`name`	Name of the new cohort table created in the cdm object.
`.softValidation`	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table with any cohort entries outside of the date range dropped

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor(nPerson = 100)
cdm$cohort1 |>
  requireInDateRange(indexDate = "cohort_start_date",
                     dateRange = as.Date(c("2010-01-01", "2019-01-01")))
```

`requireIsEntry` *Restrict cohort to specific entry*

Description

`requireIsFirstEntry()` filters cohort records, keeping only the first cohort entry per person.

Usage

```
requireIsEntry(
  cohort,
  entryRange,
  cohortId = NULL,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`entryRange`	Range for entries to include.
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`name`	Name of the new cohort table created in the cdm object.
`.softValidation`	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

A cohort table in a cdm reference.

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
cdm$cohort1 <- requireIsEntry(cdm$cohort1, c(1, Inf))
```

`requireIsFirstEntry` *Restrict cohort to first entry*

Description

`requireIsFirstEntry()` filters cohort records, keeping only the first cohort entry per person.

Usage

```
requireIsFirstEntry(
  cohort,
  cohortId = NULL,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`name`	Name of the new cohort table created in the cdm object.
`.softValidation`	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

A cohort table in a cdm reference.

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
cdm$cohort1 <- requireIsFirstEntry(cdm$cohort1)
```

`requireIsLastEntry` *Restrict cohort to last entry per person*

Description

`requireIsLastEntry()` filters cohort records, keeping only the last cohort entry per person.

Usage

```
requireIsLastEntry(
  cohort,
  cohortId = NULL,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`name`	Name of the new cohort table created in the cdm object.
`.softValidation`	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

A cohort table in a cdm reference.

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
cdm$cohort1 <- requireIsLastEntry(cdm$cohort1)
```

`requireMinCohortCount` *Filter cohorts to keep only records for those with a minimum amount of subjects*

Description

`requireMinCohortCount()` filters an existing cohort table, keeping only records from cohorts with a minimum number of individuals

Usage

```
requireMinCohortCount(
  cohort,
  minCohortCount,
  cohortId = NULL,
  name = tableName(cohort)
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`minCohortCount`	The minimum count of sbjects for a cohort to be included.
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`name`	Name of the new cohort table created in the cdm object.

Value

Cohort table

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor(nPerson = 100)

cdm$cohort1 |>
requireMinCohortCount(5)
```

`requirePriorObservation` *Restrict cohort on prior observation*

Description

`requirePriorObservation()` filters cohort records, keeping only records where individuals satisfy the specified prior observation criteria.

Usage

```
requirePriorObservation(
  cohort,
  minPriorObservation,
  cohortId = NULL,
  indexDate = "cohort_start_date",
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`minPriorObservation`	A minimum number of continuous prior observation days in the database.
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`indexDate`	Variable in cohort that contains the date to compute the demographics characteristics on which to restrict on.
`name`	Name of the new cohort table created in the cdm object.
`.softValidation`	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table with only records for individuals satisfying the prior observation requirement

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
cdm$cohort1 |>
  requirePriorObservation(indexDate = "cohort_start_date",
                          minPriorObservation = 365)
```

`requireSex` *Restrict cohort on sex*

Description

`requireSex()` filters cohort records, keeping only records where individuals satisfy the specified sex criteria.

Usage

```
requireSex(
  cohort,
  sex,
  cohortId = NULL,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`sex`	Can be "Both", "Male" or "Female".
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`name`	Name of the new cohort table created in the cdm object.
`.softValidation`	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table with only records for individuals satisfying the sex requirement

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
cdm$cohort1 |>
  requireSex(sex = "Female")
```

`requireTableIntersect` *Require cohort subjects are present in another clinical table*

Description

`requireTableIntersect()` filters a cohort table based on a requirement that an individual is seen (or not seen) to have a record (or no records) in a clinical table in some time window around an index date.

Usage

```
requireTableIntersect(
  cohort,
  tableName,
  window,
  intersections = c(1, Inf),
  cohortId = NULL,
  indexDate = "cohort_start_date",
  targetStartDate = startDateColumn(tableName),
  targetEndDate = endDateColumn(tableName),
  inObservation = TRUE,
  censorDate = NULL,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`tableName`	Name of the table to check for intersect.
`window`	A list of vectors specifying minimum and maximum days from `indexDate` to consider events over.
`intersections`	A range indicating number of intersections for criteria to be fulfilled. If a single number is passed, the number of intersections must match this.
`cohortId`	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
`indexDate`	Name of the column in the cohort that contains the date to compute the intersection.

`targetStartDate`
: Start date of reference in cohort table.

`targetEndDate`
: End date of reference in cohort table. If NULL, incidence of target event in the window will be considered as intersection, otherwise prevalence of that event will be used as intersection (overlap between cohort and event).

`inObservation`
: If TRUE only records inside an observation period will be considered.

`censorDate`
: Whether to censor overlap events at a specific date or a column date of the cohort.

`name`
: Name of the new cohort table created in the cdm object.

`.softValidation`
: Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

Cohort table with only those in the other table kept (or those that are not in the table if negate = TRUE)

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor(drugExposure = TRUE)
cdm$cohort1 |>
  requireTableIntersect(tableName = "drug_exposure",
                            indexDate = "cohort_start_date",
                            window = c(-Inf, 0))
```

---

sampleCohorts *Sample a cohort table for a given number of individuals.*

Description

`sampleCohorts()` samples an existing cohort table for a given number of people. All records of these individuals are preserved.

Usage

```
sampleCohorts(cohort, n, cohortId = NULL, name = tableName(cohort))
```

Arguments

cohort	A cohort table in a cdm reference.
n	Number of people to be sampled for each included cohort.
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
name	Name of the new cohort table created in the cdm object.

Value

Cohort table with the specified cohorts sampled.

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor(nPerson = 100)

cdm$cohort2 |> sampleCohorts(cohortId = 1, n = 10)
```

---

stratifyCohorts *Create a new cohort table from stratifying an existing one*

Description

`stratifyCohorts()` creates new cohorts, splitting an existing cohort based on specified columns on which to stratify on.

Usage

```
stratifyCohorts(
  cohort,
  strata,
  cohortId = NULL,
  removeStrata = TRUE,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

cohort	A cohort table in a cdm reference.
strata	A strata list that point to columns in cohort table.
cohortId	Vector identifying which cohorts to include (cohort_definition_id or cohort_name). Cohorts not included will be removed from the cohort set.

removeStrata Whether to remove strata columns from final cohort table.

name Name of the new cohort table created in the cdm object.

.softValidation Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

Cohort table stratified.

Examples

```
library(CohortConstructor)
library(PatientProfiles)

cdm <- mockCohortConstructor()

cdm$my_cohort <- cdm$cohort1 |>
  addAge(ageGroup = list("child" = c(0, 17), "adult" = c(18, Inf))) |>
  addSex(name = "my_cohort") |>
  stratifyCohorts(
    strata = list("sex", c("sex", "age_group")), name = "my_cohort"
  )

cdm$my_cohort

settings(cdm$my_cohort)

attrition(cdm$my_cohort)
```

---

subsetCohorts *Generate a cohort table keeping a subset of cohorts.*

Description

subsetCohorts() filters an existing cohort table, keeping only the records from cohorts that are specified.

Usage

```
subsetCohorts(
  cohort,
  cohortId,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

cohort	A cohort table in a cdm reference.
cohortId	Vector identifying which cohorts to include (cohort_definition_id or cohort_name). Cohorts not included will be removed from the cohort set.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

Cohort table with only cohorts in cohortId.

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor(nPerson = 100)

cdm$cohort1 |> subsetCohorts(cohortId = 1)
```

---

trimDemographics *Trim cohort on patient demographics*

Description

`trimDemographics()` resets the cohort start and end date based on the specified demographic criteria is satisfied.

Usage

```
trimDemographics(
  cohort,
  cohortId = NULL,
  ageRange = NULL,
  sex = NULL,
  minPriorObservation = NULL,
  minFutureObservation = NULL,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

cohort	A cohort table in a cdm reference.
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
ageRange	A list of vectors specifying minimum and maximum age.
sex	Can be "Both", "Male" or "Female".
minPriorObservation	A minimum number of continuous prior observation days in the database.
minFutureObservation	A minimum number of continuous future observation days in the database.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table with only records for individuals satisfying the demographic requirements

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor(nPerson = 100)

cdm$cohort1 |> trimDemographics(ageRange = list(c(10, 30)))
```

---

trimToDateRange *Trim cohort dates to be within a date range*

Description

`trimToDateRange()` resets the cohort start and end date based on the specified date range.

Usage

```
trimToDateRange(
  cohort,
  dateRange,
  cohortId = NULL,
```

```
  startDate = "cohort_start_date",
  endDate = "cohort_end_date",
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

cohort	A cohort table in a cdm reference.
dateRange	A window of time during which the start and end date must have been observed.
cohortId	Vector identifying which cohorts to modify (cohort_definition_id or cohort_name). If NULL, all cohorts will be used; otherwise, only the specified cohorts will be modified, and the rest will remain unchanged.
startDate	Variable with earliest date.
endDate	Variable with latest date.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

The cohort table with record timings updated to only be within the date range. Any records with all time outside of the range will have been dropped.

Examples

```
library(CohortConstructor)
cdm <- mockCohortConstructor()
cdm$cohort1 |>
  trimToDateRange(startDate = "cohort_start_date",
                  endDate = "cohort_end_date",
                  dateRange = as.Date(c("2015-01-01",
                                        "2015-12-31")))
```

---

unionCohorts *Generate cohort from the union of different cohorts*

Description

`unionCohorts()` combines different cohort entries, with those records that overlap combined and kept. Cohort entries are when an individual was in *either* of the cohorts.

Usage

```
unionCohorts(
  cohort,
  cohortId = NULL,
  gap = 0,
  cohortName = NULL,
  keepOriginalCohorts = FALSE,
  name = tableName(cohort),
  .softValidation = TRUE
)
```

Arguments

cohort	A cohort table in a cdm reference.
cohortId	Vector identifying which cohorts to include (cohort_definition_id or cohort_name). Cohorts not included will be removed from the cohort set.
gap	Number of days between two subsequent cohort entries to be merged in a single cohort record.
cohortName	Name of the returned cohort. If NULL, the cohort name will be created by collapsing the individual cohort names, separated by "_".
keepOriginalCohorts	If TRUE the original cohorts will be return together with the new ones. If FALSE only the new cohort will be returned.
name	Name of the new cohort table created in the cdm object.
.softValidation	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

A cohort table.

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor(nPerson = 100)

cdm$cohort2 <- cdm$cohort2 |> unionCohorts()
settings(cdm$cohort2)
```

`yearCohorts` *Generate a new cohort table restricting cohort entries to certain years*

Description

`yearCohorts()` splits a cohort into multiple cohorts, one for each year.

Usage

```
yearCohorts(
  cohort,
  years,
  cohortId = NULL,
  name = tableName(cohort),
  .softValidation = FALSE
)
```

Arguments

`cohort`	A cohort table in a cdm reference.
`years`	Numeric vector of years to use to restrict observation to.
`cohortId`	Vector identifying which cohorts to include (cohort_definition_id or cohort_name). Cohorts not included will be removed from the cohort set.
`name`	Name of the new cohort table created in the cdm object.
`.softValidation`	Whether to perform a soft validation of consistency. If set to FALSE four additional checks will be performed: 1) a check that cohort end date is not before cohort start date, 2) a check that there are no missing values in required columns, 3) a check that cohort duration is all within observation period, and 4) that there are no overlapping cohort entries

Value

A cohort table.

Examples

```
library(CohortConstructor)

cdm <- mockCohortConstructor(nPerson = 100)

cdm$cohort1 <- cdm$cohort1 |> yearCohorts(years = 2000:2002)
settings(cdm$cohort1)
```

Index