

Package ‘CohortAlgebra’

January 10, 2023

Type Package

Title Cohort Algebra to create new cohort(s) from existing cohorts

Version 0.4.3

Date 2023-01-10

Maintainer Gowtham Rao <rao@ohdsi.org>

Description An R package that creates new cohort(s) from previously instantiated cohorts.

Depends DatabaseConnector (>= 5.0.0),
R (>= 4.1.0)

Imports checkmate,
clock,
CohortGenerator,
dplyr,
lifecycle,
ParallelLogger,
rlang,
SqlRender

Suggests Eunomia,
remotes,
rmarkdown,
knitr,
testthat,
withr

Remotes ohdsi/CohortGenerator,
ohdsi/Eunomia,
ohdsi/ParallelLogger

License Apache License

RoxygenNote 7.2.2

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

Encoding UTF-8

Language en-US

URL <https://ohdsi.github.io/CohortAlgebra/>, <https://github.com/OHDSI/CohortAlgebra>

BugReports <https://github.com/OHDSI/CohortAlgebra/issues>

R topics documented:

copyCohorts	2
copyCohortsToTempTable	3
deleteCohort	4
eraFyCohorts	5
generateBaseCohorts	6
getBaseCohortDefinitionSet	8
getCohortIdsInCohortTable	8
intersectCohorts	9
keepCohortOverlaps	10
minusCohorts	12
modifyCohort	13
removeOverlappingSubjects	15
unionCohorts	17

Index	19
--------------	-----------

copyCohorts	<i>Copy cohorts from one table to another</i>
-------------	---

Description

Copy cohorts from one table to another table.

[Stable]

Usage

```
copyCohorts(
  connectionDetails = NULL,
  connection = NULL,
  oldToNewCohortId,
  sourceCohortDatabaseSchema = NULL,
  targetCohortDatabaseSchema = sourceCohortDatabaseSchema,
  sourceCohortTable,
  targetCohortTable,
  purgeConflicts = FALSE,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema")
)
```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.

oldToNewCohortId	A data.frame object with two columns. oldCohortId and newCohortId. Both should be integers. The oldCohortId are the cohorts that are the input cohorts that need to be transformed. The newCohortId are the cohortIds of the corresponding output after transformation. If the oldCohortId = newCohortId then the data corresponding to oldCohortId will be replaced by the data from the newCohortId.
sourceCohortDatabaseSchema	The database schema of the source cohort table.
targetCohortDatabaseSchema	The database schema of the source cohort table.
sourceCohortTable	The name of the source cohort table.
targetCohortTable	The name of the target cohort table.
purgeConflicts	If there are conflicts in the target cohort table i.e. the target cohort table already has records with newCohortId, do you want to purge and replace them with transformed. By default - it will not be replaced, and an error message is thrown.
tempEmulationSchema	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

Examples

```
## Not run:
CohortAlgebra::copyCohorts(
  connection = connection,
  sourceCohortDatabaseSchema = cohortDatabaseSchema,
  targetCohortDatabaseSchema = cohortDatabaseSchema,
  sourceCohortTable = tableName,
  targetCohortTable = tableName,
  purgeConflicts = TRUE
)

## End(Not run)
```

copyCohortsToTempTable

Copy cohorts to temp table

Description

Copy cohorts to temp table. This function is not exported.

[Stable]

Usage

```
copyCohortsToTempTable(
  connection = NULL,
  oldToNewCohortId,
  sourceCohortDatabaseSchema = NULL,
  sourceCohortTable,
  targetCohortTable = "#cohort_rows",
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema")
)
```

Arguments

connection An object of type connection as created using the [connect](#) function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.

oldToNewCohortId A data.frame object with two columns. oldCohortId and newCohortId. Both should be integers. The oldCohortId are the cohorts that are the input cohorts that need to be transformed. The newCohortId are the cohortIds of the corresponding output after transformation. If the oldCohortId = newCohortId then the data corresponding to oldCohortId will be replaced by the data from the newCohortId.

sourceCohortDatabaseSchema The database schema of the source cohort table.

sourceCohortTable The name of the source cohort table.

targetCohortTable A temp table to copy the cohorts from the source table.

tempEmulationSchema Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

deleteCohort

Delete cohort

Description

Delete all records for a given set of cohorts from the cohort table. Edit privileges to the cohort table is required.

[Stable]

Usage

```
deleteCohort(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema,
  cohortTable = "cohort",
```

```

    tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
    cohortIds
)

```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
cohortDatabaseSchema	Schema name where your cohort tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
cohortTable	The name of the cohort table.
tempEmulationSchema	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.
cohortIds	A vector of one or more Cohort Ids.

eraFyCohorts	<i>Era-fy cohort(s)</i>
--------------	-------------------------

Description

Given a table with cohort_definition_id, subject_id, cohort_start_date, cohort_end_date execute era logic. This will delete and replace the original rows with the cohort_definition_id(s). edit privileges to the cohort table is required.

[Stable]

Usage

```

eraFyCohorts(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema = NULL,
  cohortTable = "cohort",
  oldToNewCohortId,
  eraconstructorpad = 0,
  cdmDatabaseSchema = NULL,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
  purgeConflicts = FALSE
)

```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
cohortDatabaseSchema	Schema name where your cohort tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
cohortTable	The name of the cohort table.
oldToNewCohortId	A data.frame object with two columns. oldCohortId and newCohortId. Both should be integers. The oldCohortId are the cohorts that are the input cohorts that need to be transformed. The newCohortId are the cohortIds of the corresponding output after transformation. If the oldCohortId = newCohortId then the data corresponding to oldCohortId will be replaced by the data from the newCohortId.
eraconstructorpad	Optional value to pad cohort era construction logic. Default = 0. i.e. no padding.
cdmDatabaseSchema	Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.
tempEmulationSchema	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.
purgeConflicts	If there are conflicts in the target cohort table i.e. the target cohort table already has records with newCohortId, do you want to purge and replace them with transformed. By default - it will not be replaced, and an error message is thrown.

generateBaseCohorts	<i>Generate Base Cohorts</i>
---------------------	------------------------------

Description

Generates a set of cohorts that are commonly used in cohort algebra functions. Four cohorts will be generated with the cohort_definition_id of 0, -1, -2, -3 for Observation Period, Visits all, Visits Inpatient, Visits Emergency Room.

[Experimental]

Usage

```
generateBaseCohorts(
  connectionDetails = NULL,
  cohortDatabaseSchema,
  cdmDatabaseSchema,
  cohortTable = "CohortsBase",
  incremental,
  incrementalFolder = NULL,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema")
)
```

Arguments

connectionDetails
An object of type `connectionDetails` as created using the [createConnectionDetails](#) function in the `DatabaseConnector` package. Can be left `NULL` if connection is provided.

cohortDatabaseSchema
Schema name where your cohort tables reside. Note that for SQL Server, this should include both the database and schema name, for example `'scratch.dbo'`.

cdmDatabaseSchema
Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example `'cdm_data.dbo'`.

cohortTable
The name of the cohort table.

incremental
Create only cohorts that haven't been created before?

incrementalFolder
If `incremental = TRUE`, specify a folder where records are kept of which definition has been executed.

tempEmulationSchema
Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

Examples

```
## Not run:
CohortAlgebra::generateBaseCohorts(
  connection = connection,
  cohortDatabaseSchema = cohortDatabaseSchema,
  cdmDatabaseSchema = cdmDatabaseSchema,
  cohortTable = tableName,
  incremental = TRUE,
  incrementalFolder = incrementalFolder
)

## End(Not run)
```

```
getBaseCohortDefinitionSet
```

Base cohort, cohort definition set.

Description

Base cohort, cohort definition set.

Usage

```
getBaseCohortDefinitionSet()
```

```
getCohortIdsInCohortTable
```

Get cohort ids in table

Description

Get cohort ids in table

[Stable]

Usage

```
getCohortIdsInCohortTable(
  connection = NULL,
  cohortDatabaseSchema = NULL,
  cohortTable,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema")
)
```

Arguments

connection An object of type connection as created using the [connect](#) function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.

cohortDatabaseSchema Schema name where your cohort tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

cohortTable The name of the cohort table.

tempEmulationSchema Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

intersectCohorts	<i>Intersect cohort(s)</i>
------------------	----------------------------

Description

Find the common cohort period for persons present in all the cohorts. Note: if subject is not found in any of the cohorts, then they will not be in the final cohort.

[Stable]

Usage

```
intersectCohorts(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema = NULL,
  cohortTable = "cohort",
  cohortIds,
  newCohortId,
  purgeConflicts = FALSE,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema")
)
```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
cohortDatabaseSchema	Schema name where your cohort tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
cohortTable	The name of the cohort table.
cohortIds	A vector of one or more Cohort Ids.
newCohortId	The cohort id of the result cohort.
purgeConflicts	If there are conflicts in the target cohort table i.e. the target cohort table already has records with newCohortId, do you want to purge and replace them with transformed. By default - it will not be replaced, and an error message is thrown.
tempEmulationSchema	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

Examples

```
## Not run:
intersectCohorts(
  connectionDetails = Eunomia::getEunomiaConnectionDetails(),
  cohortDatabaseSchema = "main",
  cohortTable = "cohort",
  cohortIds = c(1, 2, 3),
  newCohortId = 9,
  purgeConflicts = TRUE
)

## End(Not run)
```

keepCohortOverlaps	<i>Keep records in cohort that overlap with another cohort</i>
--------------------	--

Description

Keep records in cohort that overlap with another cohort. Given a Cohort A, check if the records of subjects in cohort A overlaps with records for the same subject in cohort B. If there is overlap then only keep those records in Cohort A. All non overlapping records in Cohort A will be removed. Overlap is defined as $b.cohort_end_date \geq a.cohort_start_date$ AND $b.cohort_start_date \leq a.cohort_end_date$. The overlap logic maybe offset by using a `startDayOffset` (applied on cohort A's `cohort_start_date`) and `endDayOffset` (applied on Cohort A's `cohort_end_date`). If while applying offset, the window becomes such that $(a.cohort_start_date + startDayOffset) > (a.cohort_end_date + endDayOffset)$ that record is ignored and thus deleted.

By default we are looking for atleast one day of overlap. We can change this to look for any number of overlap days e.g. 2 days of overlap in the window. The overlap days are calculated as the total number of days between maximum of `cohort_start_date`'s of both cohorts, and minimum of `cohort_end_date`'s of both cohorts, using offset when used.

Overlap formula is $(\min(a.cohort_end_date, b.cohort_end_date) - \max(a.cohort_start_date, b.cohort_start_date)) + 1$. Note the use of +1, i.e. the lowest number of days of overlap is 1 day.

[Experimental]

Usage

```
keepCohortOverlaps(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema = NULL,
  cohortTable = "cohort",
  firstCohortId,
  secondCohortId,
  newCohortId,
  offsetCohortStartDate = 0,
  offsetCohortEndDate = 0,
  restrictSecondCohortStartBeforeFirstCohortStart = FALSE,
  restrictSecondCohortStartAfterFirstCohortStart = FALSE,
  minimumOverlapDays = 1,
  purgeConflicts = FALSE,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema")
)
```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
cohortDatabaseSchema	Schema name where your cohort tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
cohortTable	The name of the cohort table.
firstCohortId	The cohort id of the cohort whose records will be retained after the operation.
secondCohortId	The cohort id of the cohort that will be used to check for the presence of overlap.
newCohortId	The cohort id of the result cohort.
offsetCohortStartDate	(Default = 0) If you want to offset cohort start date, please provide a integer number.
offsetCohortEndDate	(Default = 0) If you want to offset cohort start date, please provide a integer number.
restrictSecondCohortStartBeforeFirstCohortStart	(Default = FALSE) If TRUE, then the secondCohort's cohort_start_date should be < firstCohort's cohort_start_date.
restrictSecondCohortStartAfterFirstCohortStart	(Default = FALSE) If TRUE, then the secondCohort's cohort_start_date should be > firstCohort's cohort_start_date.
minimumOverlapDays	(Default = 1) The minimum number of days of overlap.
purgeConflicts	If there are conflicts in the target cohort table i.e. the target cohort table already has records with newCohortId, do you want to purge and replace them with transformed. By default - it will not be replaced, and an error message is thrown.
tempEmulationSchema	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

Examples

```
## Not run:
keepCohortOverlaps(
  connectionDetails = Eunomia::getEunomiaConnectionDetails(),
  cohortDatabaseSchema = "main",
  cohortTable = "cohort",
  firstCohortId = 1,
  secondCohortId = 2,
  newCohortId = 9,
  purgeConflicts = TRUE
)
```

```
## End(Not run)
```

minusCohorts	<i>Minus cohort(s)</i>
--------------	------------------------

Description

Given two cohorts, subtract (minus) the dates from the first cohort, the dates the subject also had on the second cohort.

[Stable]

Usage

```
minusCohorts(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema = NULL,
  cohortTable = "cohort",
  firstCohortId,
  secondCohortId,
  newCohortId,
  purgeConflicts = FALSE,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema")
)
```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
cohortDatabaseSchema	Schema name where your cohort tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
cohortTable	The name of the cohort table.
firstCohortId	The cohort id of the cohort from which to subtract.
secondCohortId	The cohort id of the cohort that is used to subtract.
newCohortId	The cohort id of the result cohort.
purgeConflicts	If there are conflicts in the target cohort table i.e. the target cohort table already has records with newCohortId, do you want to purge and replace them with transformed. By default - it will not be replaced, and an error message is thrown.
tempEmulationSchema	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

Examples

```
## Not run:
minusCohorts(
  connectionDetails = Eunomia::getEunomiaConnectionDetails(),
  cohortDatabaseSchema = "main",
  cohortTable = "cohort",
  firstCohortId = 1,
  secondCohortId = 2,
  newCohortId = 9,
  purgeConflicts = TRUE
)

## End(Not run)
```

 modifyCohort

Modify cohort

Description

Modify cohort by censoring, padding, limiting cohorts periods. Censoring: Provide a date for right, left, both censoring. All cohorts will be truncated to the given date. Pad days: Add days to either cohort start or cohort end dates. Maybe negative numbers. Final cohort will not be outside the persons observation period. Limit cohort periods: Filter the cohorts to a given date range of cohort start, or cohort end or both.

cdmDataschema is required when eraConstructorPad is > 0. eraConstructorPad is optional. It is also required when checking for minimum continuous prior or post observation period.

[Experimental]

Usage

```
modifyCohort(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema = NULL,
  cdmDatabaseSchema = NULL,
  cohortTable = "cohort",
  oldCohortId,
  newCohortId = oldCohortId,
  cohortStartCensorDate = NULL,
  cohortEndCensorDate = NULL,
  cohortStartFilterRange = NULL,
  cohortEndFilterRange = NULL,
  cohortStartPadDays = NULL,
  cohortEndPadDays = NULL,
  filterGenderConceptId = NULL,
  filterByAgeRange = NULL,
  firstOccurrence = FALSE,
  filterByMinimumCohortPeriod = NULL,
  filterByMinimumPriorObservationPeriod = NULL,
  filterByMinimumPostObservationPeriod = NULL,
```

```

tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
purgeConflicts = TRUE
)

```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
cohortDatabaseSchema	Schema name where your cohort tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
cdmDatabaseSchema	Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.
cohortTable	The name of the cohort table.
oldCohortId	The cohort id of the cohort that needs to be modified.
newCohortId	The cohort id of the result cohort.
cohortStartCensorDate	the minimum date for the cohort. All rows with cohort start date before this date will be censored to given date.
cohortEndCensorDate	the maximum date for the cohort. All rows with cohort end date after this date will be censored to given date.
cohortStartFilterRange	A range of dates representing minimum to maximum to filter the cohort by its cohort start date e.g c(as.Date('1999-01-01'), as.Date('1999-12-31'))
cohortEndFilterRange	A range of dates representing minimum to maximum to filter the cohort by its cohort end date e.g c(as.Date('1999-01-01'), as.Date('1999-12-31'))
cohortStartPadDays	An integer value to pad the cohort start date. Default is 0 - no padding. The final cohort will have no days outside the observation period dates of the initial observation period. If negative padding, then cohortStartDate will not shift to before corresponding observationPeriodStartDate, it will be forced to be equal to observationPeriodStartDate. If positive padding, then cohortStartDate will not shift beyond observationPeriodEndDate, it will be forced to be equal to observationPeriodEndDate. Also cohortStartDate will not be more than cohortEndDate - it will be forced to be equal to cohortEndDate.
cohortEndPadDays	An integer value to pad the cohort start date. Default is 0 - no padding. The final cohort will have no days outside the observation period dates of the initial observation period. If negative padding, then cohortEndDate will not shift to before corresponding observationPeriodEndDate, it will be forced to be equal to observationPeriodEndDate. If positive padding, then cohortEndDate will not shift

beyond observationPeriodStartDate, it will be forced to be equal to observationPeriodStartDate. Also cohortEndDate will not be less than cohortStartDate - it will be forced to be equal to cohortStartDate.

filterGenderConceptId

Provide an array of integers corresponding to conceptId to look for in the gender_concept_id field of the person table.

filterByAgeRange

Provide an array of two values, where second value is \geq first value to filter the persons age on cohort_start_date. Age is calculated as $\text{YEAR}(\text{cohort_start_date}) - \text{person.year_of_birth}$

firstOccurrence

Do you want to restrict the cohort to the first occurrence per person?

filterByMinimumCohortPeriod

Do you want to filter cohort records by minimum cohort period, i.e. cohort period is calculated as $\text{DATEDIFF}(\text{cohort_start_date}, \text{cohort_start_date})$. if cohort_start_date = cohort_end_date then days = 0

filterByMinimumPriorObservationPeriod

Do you want to filter cohort records by minimum Prior continuous Observation period

filterByMinimumPostObservationPeriod

Do you want to filter cohort records by minimum Post continuous Observation period

tempEmulationSchema

Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

purgeConflicts If there are conflicts in the target cohort table i.e. the target cohort table already has records with newCohortId, do you want to purge and replace them with transformed. By default - it will not be replaced, and an error message is thrown.

Examples

```
## Not run:
CohortAlgebra::modifyCohort(
  connection = connection,
  cohortDatabaseSchema = cohortDatabaseSchema,
  cohortTable = tableName,
  oldCohortId = 3,
  newCohortId = 2,
  cohortEndFilterRange = c(as.Date("2010-01-01"), as.Date("2010-01-09")),
  purgeConflicts = TRUE
)

## End(Not run)
```

removeOverlappingSubjects

Remove subjects in cohort that overlap with another cohort

Description

Remove subjects in cohort that overlap with another cohort. Given a Cohort A, check if the records of subjects in cohort A overlaps with records for the same subject in cohort B. If there is overlap then remove all records of that subject from Cohort A. Overlap is defined as $b.cohort_end_date \geq a.cohort_start_date$ AND $b.cohort_start_date \leq a.cohort_end_date$. The overlap logic maybe offset by using a startDayOffset (applied on cohort A's cohort_start_date) and endDayOffset (applied on Cohort A's cohort_end_date). If while applying offset, the window becomes such that $(a.cohort_start_date + startDayOffset) > (a.cohort_end_date + endDayOffset)$ that record is ignored and thus deleted.

[Experimental]

Usage

```
removeOverlappingSubjects(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema,
  cohortId,
  newCohortId,
  cohortsWithSubjectsToRemove,
  offsetCohortStartDate = -99999,
  offsetCohortEndDate = 99999,
  restrictSecondCohortStartBeforeFirstCohortStart = FALSE,
  restrictSecondCohortStartAfterFirstCohortStart = FALSE,
  cohortTable = "cohort",
  purgeConflicts = FALSE,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema")
)
```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
cohortDatabaseSchema	Schema name where your cohort tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
cohortId	The cohort id of the cohort whose subjects will be removed.
newCohortId	The cohort id of the result cohort.
cohortsWithSubjectsToRemove	An array of one or more cohorts with subjects to remove from given cohorts.
offsetCohortStartDate	(Default = 0) If you want to offset cohort start date, please provide a integer number.
offsetCohortEndDate	(Default = 0) If you want to offset cohort start date, please provide a integer number.

`restrictSecondCohortStartBeforeFirstCohortStart`
 (Default = FALSE) If TRUE, then the secondCohort's cohort_start_date should be < firstCohort's cohort_start_date.

`restrictSecondCohortStartAfterFirstCohortStart`
 (Default = FALSE) If TRUE, then the secondCohort's cohort_start_date should be > firstCohort's cohort_start_date.

`cohortTable` The name of the cohort table.

`purgeConflicts` If there are conflicts in the target cohort table i.e. the target cohort table already has records with newCohortId, do you want to purge and replace them with transformed. By default - it will not be replaced, and an error message is thrown.

`tempEmulationSchema`
 Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

Examples

```
## Not run:
removeOverlappingSubjects(
  connection = connection,
  cohortDatabaseSchema = cohortDatabaseSchema,
  cohortId = 1,
  newCohortId = 9,
  cohortsWithSubjectsToRemove = c(3),
  purgeConflicts = FALSE,
  cohortTable = tableName
)

## End(Not run)
```

unionCohorts	<i>Union cohort(s)</i>
--------------	------------------------

Description

Given a specified array of cohortIds in a cohort table, perform cohort union operator to create new cohorts.

[Stable]

Usage

```
unionCohorts(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema = NULL,
  cohortTable = "cohort",
  oldToNewCohortId,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
  purgeConflicts = FALSE
)
```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
cohortDatabaseSchema	Schema name where your cohort tables reside. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
cohortTable	The name of the cohort table.
oldToNewCohortId	A data.frame object with two columns. oldCohortId and newCohortId. Both should be integers. The oldCohortId are the cohorts that are the input cohorts that need to be transformed. The newCohortId are the cohortIds of the corresponding output after transformation. If the oldCohortId = newCohortId then the data corresponding to oldCohortId will be replaced by the data from the newCohortId.
tempEmulationSchema	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.
purgeConflicts	If there are conflicts in the target cohort table i.e. the target cohort table already has records with newCohortId, do you want to purge and replace them with transformed. By default - it will not be replaced, and an error message is thrown.

Examples

```
## Not run:
unionCohorts(
  connectionDetails = Eunomia::getEunomiaConnectionDetails(),
  cohortDatabaseSchema = "main",
  cohortTable = "cohort",
  oldToNewCohortId = dplyr::tibble(
    oldCohortId = c(1, 2, 3),
    newCohortId = c(9, 9, 9)
  ),
  purgeConflicts = TRUE
)

## End(Not run)
```

Index

connect, [2](#), [4–6](#), [8](#), [9](#), [11](#), [12](#), [14](#), [16](#), [18](#)
copyCohorts, [2](#)
copyCohortsToTempTable, [3](#)
createConnectionDetails, [2](#), [5–7](#), [9](#), [11](#), [12](#),
[14](#), [16](#), [18](#)

deleteCohort, [4](#)

eraFyCohorts, [5](#)

generateBaseCohorts, [6](#)
getBaseCohortDefinitionSet, [8](#)
getCohortIdsInCohortTable, [8](#)

intersectCohorts, [9](#)

keepCohortOverlaps, [10](#)

minusCohorts, [12](#)
modifyCohort, [13](#)

removeOverlappingSubjects, [15](#)

unionCohorts, [17](#)