Package: Characterization (via r-universe)

October 16, 2026

Type Package

Title Implement Descriptive Studies Using the Common Data Model

Version 4.1.0

Date 2026-9-15

Maintainer Jenna Reps <jreps@its.jnj.com>

Description An end-to-end framework that enables users to implement various descriptive studies for a given set of target and outcome cohorts for data mapped to the Observational Medical Outcomes Partnership Common Data Model.

License Apache License 2.0

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

BugReports https://github.com/OHDSI/Characterization/issues

Depends R (>= 4.0.0)

Imports Andromeda (>= 1.0.0), DatabaseConnector (>= 6.3.1), digest, FeatureExtraction (>= 3.13.0), SqlRender (>= 1.9.0), ParallelLogger (>= 3.0.0), ResultModelManager, checkmate, dplyr, readr, rlang

Suggests CohortIncidence (>= 4.1.1), devtools, formatR, testthat, kableExtra, knitr, markdown, rmarkdown, OhdsiShinyAppBuilder, shiny, withr

NeedsCompilation no

Encoding UTF-8

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

Author Jenna Reps [aut, cre], Patrick Ryan [aut], Chris Knoll [aut]

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

Date/Publication 2026-09-16 05:20:02 UTC

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

RemoteRef HEAD

RemoteSha f285cdf4cf90d6b238ccaf09c19c75bd2c41af40

Contents

cleanIncremental . . . 2
cleanNonIncremental . . . 3
createCaseSeriesSettings . . . 4
createCharacterizationSettings . . . 5
createCharacterizationTables . . . 7
createCohortIncidenceSettings . . . 8
createDechallengeRechallengeSettings . . . 9
createDuringCovariateSettings . . . 10
createRiskFactorSettings . . . 13
createSqliteDatabase . . . 14
createStudyPopulationSettings . . . 15
createTargetBaselineSettings . . . 16
createTimeToEventSettings . . . 17
exampleOmopConnectionDetails . . . 18
getDbDuringCovariateData . . . 19
insertResultsToDatabase . . . 21
loadCharacterizationSettings . . . 23
runCharacterizationAnalyses . . . 24
saveCharacterizationSettings . . . 27
viewCharacterization . . . 28

Index **30**

`cleanIncremental` *Removes csv files from folders that have not been marked as completed and removes the record of the execution file*

Description

Removes csv files from folders that have not been marked as completed and removes the record of the execution file

Usage

```
cleanIncremental(executionFolder, ignoreWhenEmpty = FALSE)
```

Arguments

`executionFolder`
: The folder that has the execution files

`ignoreWhenEmpty`
: When TRUE, if there are no incremental logs then nothing is run

Value

A list with the settings

See Also

Other Incremental: `cleanNonIncremental()`

Examples

```
cleanIncremental(
  file.path(tempdir(), 'incremental'),
  ignoreWhenEmpty = TRUE
)
```

`cleanNonIncremental` *Removes csv files from the execution folder as there should be no csv files when running in non-incremental model*

Description

Removes csv files from the execution folder as there should be no csv files when running in non-incremental model

Usage

```
cleanNonIncremental(executionFolder)
```

Arguments

`executionFolder`
: The folder that has the execution files

Value

A list with the settings

See Also

Other Incremental: `cleanIncremental()`

Examples

```
# example code

cleanNonIncremental(file.path(tempdir(), 'incremental'))
```

createCaseSeriesSettings *Create aggregate covariate study settings*

Description

Create aggregate covariate study settings

Usage

```
createCaseSeriesSettings(
  studyPopulationSettings,
  outcomeIds,
  outcomeWashoutDays = 0,
  riskWindowStart = 1,
  startAnchor = "cohort start",
  riskWindowEnd = 365,
  endAnchor = "cohort start",
 caseCovariateSettings = createDuringCovariateSettings(useConditionGroupEraDuring =
    TRUE, useDrugGroupEraDuring = TRUE, useProcedureOccurrenceDuring = TRUE,
   useDeviceExposureDuring = TRUE, useMeasurementDuring = TRUE, useObservationDuring =
    TRUE, useVisitConceptCountDuring = TRUE),
  casePreTargetDuration = 365,
  casePostOutcomeDuration = 365
)
```

Arguments

studyPopulationSettings	A List of object created using createStudyPopulationSettings that specifies target cohorts and inclusion criteria
outcomeIds	A list of cohortIds for the outcome cohorts
outcomeWashoutDays	A single integer value. Patients with the outcome within outcomeWashout days prior to index are excluded from the risk factor analysis
riskWindowStart	The start of the risk window (in days) relative to the ‘startAnchor‘.
startAnchor	The anchor point for the start of the risk window. Can be ‘"cohort start"‘ or ‘"cohort end"‘.
riskWindowEnd	The end of the risk window (in days) relative to the ‘endAnchor‘.
endAnchor	The anchor point for the end of the risk window. Can be ‘"cohort start"‘ or ‘"cohort end"‘.
caseCovariateSettings	An object created using createDuringCovariateSettings

casePreTargetDuration
: A single integer value. The number of days prior to case index we use for FeatureExtraction

casePostOutcomeDuration
: A single integer value. The number of days prior to case index we use for FeatureExtraction

Value

A list with the settings

See Also

Other Aggregate: createRiskFactorSettings(), createTargetBaselineSettings()

Examples

```
caseSeriesSetting <- createCaseSeriesSettings(
  studyPopulationSettings = createStudyPopulationSettings(
    targetIds = c(1,2),
    minPriorObservation = 365,
    limitToFirstInNDays = 365
  ),
  outcomeIds = c(3),
  outcomeWashoutDays = 90,
  riskWindowStart = 1,
  startAnchor = "cohort start",
  riskWindowEnd = 365,
  endAnchor = "cohort start",
  casePreTargetDuration = 365,
  casePostOutcomeDuration = 365
)
```

---

createCharacterizationSettings

Create the settings for a large scale characterization study

Description

This function creates a list of settings for different characterization studies

Usage

```
createCharacterizationSettings(
  timeToEventSettings = NULL,
  dechallengeRechallengeSettings = NULL,
  targetBaselineSettings = NULL,
  riskFactorSettings = NULL,
```

```
  caseSeriesSettings = NULL,
  cohortIncidenceSettings = NULL
)
```

Arguments

`timeToEventSettings`
: A list of timeToEvent settings

`dechallengeRechallengeSettings`
: A list of dechallengeRechallenge settings

`targetBaselineSettings`
: A list of targetBaselineSettings settings

`riskFactorSettings`
: A list of riskFactorSettings settings

`caseSeriesSettings`
: A list of caseSeriesSettings settings

`cohortIncidenceSettings`
: A lsit of cohortIncidenceSettings settings

Details

Specify one or more timeToEvent, dechallengeRechallenge and aggregateCovariate settings

Value

Returns the connection to the sqlite database

See Also

Other LargeScale: `loadCharacterizationSettings()`, `runCharacterizationAnalyses()`, `saveCharacterizationSet`

Examples

```
# example code

drSet <- createDechallengeRechallengeSettings(
  studyPopulationSettings = createStudyPopulationSettings(
    targetIds = c(1,2),
    limitToFirstInNDays = 0,
    minPriorObservation = 0
    ),
  outcomeIds = 3
)

cSet <- createCharacterizationSettings(
  dechallengeRechallengeSettings = drSet
)
```

createCharacterizationTables *Create the results tables to store characterization results into a database*

Description

This function executes a large set of SQL statements to create tables that can store results

Usage

```
createCharacterizationTables(
  connectionDetails,
  resultSchema,
  targetDialect = "postgresql",
  deleteExistingTables = TRUE,
  createTables = TRUE,
  tablePrefix = "c_",
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema")
)
```

Arguments

connectionDetails	The connectionDetails to a database created by using the function `createConnectDetails` in the `DatabaseConnector` package.
resultSchema	The name of the database schema that the result tables will be created.
targetDialect	The database management system being used
deleteExistingTables	If true any existing tables matching the Characterization result tables names will be deleted
createTables	If true the Characterization result tables will be created
tablePrefix	A string appended to the Characterization result tables
tempEmulationSchema	The temp schema used when the database management system is oracle

Details

This function can be used to create (or delete) Characterization result tables

Value

Returns NULL but creates the required tables into the specified database schema.

See Also

Other Database: `createSqliteDatabase()`, `insertResultsToDatabase()`

Examples

```
# create sqlite database
charResultDbCD <- createSqliteDatabase()

# create database results tables
createCharacterizationTables(
   connectionDetails = charResultDbCD,
   resultSchema = 'main'
 )
```

`createCohortIncidenceSettings` *Create cohort incidence study settings*

Description

Create cohort incidence study settings

Usage

```
createCohortIncidenceSettings(
  studyPopulationSettings,
  outcomeIds,
  outcomeWashoutDays = 0,
  riskWindowStart = 1,
  startAnchor = "cohort start",
  riskWindowEnd = 365,
  endAnchor = "cohort start",
  byAge = FALSE,
  byGender = FALSE,
  byYear = FALSE,
  ageBreaks = NULL,
  ageBreakList = NULL,
  startDate = "",
  endDate = ""
)
```

Arguments

`studyPopulationSettings`
: An object created using `createStudyPopulationSettings` or a list of `createStudyPopulationSettin` that specifies cohort inclusion criteria

`outcomeIds`
: A vector of cohortIds for the outcome cohorts

`outcomeWashoutDays`
: A vector of integers specifying the washout days for the outcomeIds

riskWindowStart	The start of the risk window (in days) relative to the ‘startAnchor‘.
startAnchor	The anchor point for the start of the risk window. Can be ‘"cohort start"‘ or ‘"cohort end"‘.
riskWindowEnd	The end of the risk window (in days) relative to the ‘endAnchor‘.
endAnchor	The anchor point for the end of the risk window. Can be ‘"cohort start"‘ or ‘"cohort end"‘.
byAge	Whether to stratify the incidence rates by age groups (specified via ageBreaks or ageBreakList)
byGender	Whether to stratify the incidence rates by gender
byYear	Whether to stratify the incidence rates by index year
ageBreaks	a vector of integers indicating the age group bounds
ageBreakList	a list of ageBreaks, used to specify multiple age break strata.
startDate	a character vector representing a date in YYYY-MM-DD format
endDate	a character vector representing a date in YYYY-MM-DD format

Value

An list with the cohort incidence settings

Examples

```
# example code

ciSet <- createCohortIncidenceSettings(
  studyPopulationSettings = createStudyPopulationSettings(
    targetIds = c(1,2),
    limitToFirstInNDays = 0,
    minPriorObservation = 0
    ),
  outcomeIds = 3
)
```

---

createDechallengeRechallengeSettings

Create dechallenge rechallenge study settings

Description

Create dechallenge rechallenge study settings

Usage

```
createDechallengeRechallengeSettings(
  studyPopulationSettings,
  outcomeIds,
  dechallengeStopInterval = 30,
  dechallengeEvaluationWindow = 30
)
```

Arguments

studyPopulationSettings	An object created using createStudyPopulationSettings of a list of createStudyPopulationSettin that specifies cohort inclusion criteria
outcomeIds	A list of cohortIds for the outcome cohorts
dechallengeStopInterval	An integer specifying the how much time to add to the cohort_end when determining whether the event starts during cohort and ends after
dechallengeEvaluationWindow	An integer specifying the period of time after the cohort_end when you cannot see an outcome for a dechallenge success

Value

A list with the settings

Examples

```
drSet <- createDechallengeRechallengeSettings(
  studyPopulationSettings = createStudyPopulationSettings(
    targetIds = c(1,2),
    limitToFirstInNDays = 0,
    minPriorObservation = 0
    ),
  outcomeIds = 3
)
```

---

createDuringCovariateSettings

Create during covariate settings

Description

Create during covariate settings

Usage

```
createDuringCovariateSettings(
  useConditionOccurrenceDuring = FALSE,
  useConditionOccurrencePrimaryInpatientDuring = FALSE,
  useConditionEraDuring = FALSE,
  useConditionGroupEraDuring = FALSE,
  useDrugExposureDuring = FALSE,
  useDrugEraDuring = FALSE,
  useDrugGroupEraDuring = FALSE,
  useProcedureOccurrenceDuring = FALSE,
  useDeviceExposureDuring = FALSE,
  useMeasurementDuring = FALSE,
  useObservationDuring = FALSE,
  useVisitCountDuring = FALSE,
  useVisitConceptCountDuring = FALSE,
  includedCovariateConceptIds = c(),
  addDescendantsToInclude = FALSE,
  excludedCovariateConceptIds = c(),
  addDescendantsToExclude = FALSE,
  includedCovariateIds = c()
)
```

Arguments

`useConditionOccurrenceDuring`
: One covariate per condition in the condition_occurrence table starting between cohort start and cohort end. (analysis ID 109)

`useConditionOccurrencePrimaryInpatientDuring`
: One covariate per condition observed as a primary diagnosis in an inpatient setting in the condition_occurrence table starting between cohort start and cohort end. (analysis ID 110)

`useConditionEraDuring`
: One covariate per condition in the condition_era table starting between cohort start and cohort end. (analysis ID 217)

`useConditionGroupEraDuring`
: One covariate per condition era rolled up to groups in the condition_era table starting between cohort start and cohort end. (analysis ID 218)

`useDrugExposureDuring`
: One covariate per drug in the drug_exposure table between cohort start and end. (analysisId 305)

`useDrugEraDuring`
: One covariate per drug in the drug_era table between cohort start and end. (analysis ID 417)

`useDrugGroupEraDuring`
: One covariate per drug rolled up to ATC groups in the drug_era table between cohort start and end. (analysis ID 418)

useProcedureOccurrenceDuring
: One covariate per procedure in the procedure_occurrence table between cohort start and end. (analysis ID 505)

useDeviceExposureDuring
: One covariate per device in the device exposure table starting between cohort start and end. (analysis ID 605)

useMeasurementDuring
: One covariate per measurement in the measurement table between cohort start and end. (analysis ID 713)

useObservationDuring
: One covariate per observation in the observation table between cohort start and end. (analysis ID 805)

useVisitCountDuring
: The number of visits observed between cohort start and end. (analysis ID 926)

useVisitConceptCountDuring
: The number of visits observed between cohort start and end, stratified by visit concept ID. (analysis ID 927)

includedCovariateConceptIds
: A list of concept IDs that should be used to construct covariates.

addDescendantsToInclude
: Should descendant concept IDs be added to the list of concepts to include?

excludedCovariateConceptIds
: A list of concept IDs that should NOT be used to construct covariates.

addDescendantsToExclude
: Should descendant concept IDs be added to the list of concepts to exclude?

includedCovariateIds
: A list of covariate IDs that should be restricted to.

Details

creates an object specifying how during covariates should be constructed from data in the CDM model.

Value

An object of type `covariateSettings`, to be used in other functions.

See Also

Other CovariateSetting: `getDbDuringCovariateData()`

Examples

```
settings <- createDuringCovariateSettings(
  useConditionOccurrenceDuring = TRUE,
  useConditionOccurrencePrimaryInpatientDuring = FALSE,
  useConditionEraDuring = FALSE,
  useConditionGroupEraDuring = FALSE
```

```
)
```

`createRiskFactorSettings` *Create risk factor study settings*

Description

Create risk factor study settings

Usage

```
createRiskFactorSettings(
  studyPopulationSettings,
  outcomeIds,
  outcomeWashoutDays = 0,
  riskWindowStart = 1,
  startAnchor = "cohort start",
  riskWindowEnd = 365,
  endAnchor = "cohort start",
 covariateSettings = FeatureExtraction::createCovariateSettings(useDemographicsGender =
  TRUE, useDemographicsAge = TRUE, useDemographicsAgeGroup = TRUE, useDemographicsRace
   = TRUE, useDemographicsEthnicity = TRUE, useDemographicsIndexYear = TRUE,
   useDemographicsIndexMonth = TRUE, useDemographicsTimeInCohort = TRUE,
  useDemographicsPriorObservationTime = TRUE, useDemographicsPostObservationTime =
   TRUE, useConditionGroupEraLongTerm = TRUE, useDrugGroupEraOverlapping = TRUE,
   useDrugGroupEraLongTerm = TRUE, useProcedureOccurrenceLongTerm = TRUE,

   useMeasurementLongTerm = TRUE, useObservationLongTerm = TRUE,
   useDeviceExposureLongTerm = TRUE, useVisitConceptCountLongTerm = TRUE,
   useConditionGroupEraShortTerm = TRUE, useDrugGroupEraShortTerm = TRUE,
   useProcedureOccurrenceShortTerm = TRUE, useMeasurementShortTerm = TRUE,
   useObservationShortTerm = TRUE, useDeviceExposureShortTerm = TRUE,
   useVisitConceptCountShortTerm = TRUE, endDays = 0, longTermStartDays = -365,
   shortTermStartDays = -30)
)
```

Arguments

`studyPopulationSettings`	A list of objects created using `createStudyPopulationSettings` that specifies target cohorts and inclusion criteria
`outcomeIds`	A list of cohortIds for the outcome cohorts
`outcomeWashoutDays`	A single integer value. Patients with the outcome within outcomeWashout days prior to index are excluded from the risk factor analysis

riskWindowStart
: The start of the risk window (in days) relative to the ‘startAnchor‘.

startAnchor
: The anchor point for the start of the risk window. Can be ‘"cohort start"‘ or ‘"cohort end"‘.

riskWindowEnd
: The end of the risk window (in days) relative to the ‘endAnchor‘.

endAnchor
: The anchor point for the end of the risk window. Can be ‘"cohort start"‘ or ‘"cohort end"‘.

covariateSettings
: An object created using `FeatureExtraction::createCovariateSettings`

Value

A list with the settings

See Also

Other Aggregate: `createCaseSeriesSettings()`, `createTargetBaselineSettings()`

Examples

```
riskFactorSetting <- createRiskFactorSettings(
  studyPopulationSettings = createStudyPopulationSettings(
    targetIds = c(1,2),
    minPriorObservation = 365,
    limitToFirstInNDays = 99999
  ),
  outcomeIds = c(3),
  outcomeWashoutDays = 90,
  riskWindowStart = 1,
  startAnchor = "cohort start",
  riskWindowEnd = 365,
  endAnchor = "cohort start"
)
```

---

createSqliteDatabase *Create an sqlite database connection*

Description

This function creates a connection to an sqlite database

Usage

```
createSqliteDatabase(sqliteLocation = tempdir())
```

Arguments

sqliteLocation The location of the sqlite database

Details

This function creates a sqlite database and connection

Value

Returns the connection detail object to the sqlite database

See Also

Other Database: `createCharacterizationTables()`, `insertResultsToDatabase()`

Examples

```
charResultDbCD <- createSqliteDatabase()
```

---

createStudyPopulationSettings

create the study population settings

Description

create the study population settings

Usage

```
createStudyPopulationSettings(
  targetIds,
  limitToFirstInNDays = 0,
  minPriorObservation = 0,
  nestingCohortId = NULL,
  minAge = NULL,
  maxAge = NULL,
  studyStartDate = NULL,
  studyEndDate = NULL,
  genderConceptIds = NULL
)
```

Arguments

`targetIds`	A target cohort id or vector of target cohort ids to do the subsetting to
`limitToFirstInNDays`	Should only the first exposure in N days per subject be included?
`minPriorObservation`	The minimum required continuous observation time prior to index date for a person to be included in the cohort.

nestingCohortId
: A cohort definition id to restrict the target cohort. Patient in the target cohort are only included if they are also in the nesting cohort at index.

minAge
: The minimum age required to be in the target at index

maxAge
: The maximum age required to be in the target at index

studyStartDate
: The earliest date to be included into the target. Date format is 'yyyymmdd'.

studyEndDate
: The latest date to be included into the target. Date format is 'yyyymmdd'.

genderConceptIds
: A target cohort subject's gender concept to restrict to

Value

A data.frame containing all the settings required for creating the study populations of interest

See Also

Other helper: `exampleOmopConnectionDetails()`

Examples

```
# Create study population settings with a washout period of 365 days and
# restricted to adults for target dates that occur for the first time in 365 days.
populationSettings <- createStudyPopulationSettings(
   targetId  = 1,
   limitToFirstInNDays = 365,
   minPriorObservation = 365,
   minAge = 18
   )
```

---

createTargetBaselineSettings

Create target baseline aggregate covariate study settings

Description

Create target baseline aggregate covariate study settings

Usage

```
createTargetBaselineSettings(
  studyPopulationSettings,
 covariateSettings = FeatureExtraction::createCovariateSettings(useDemographicsGender =
  TRUE, useDemographicsAge = TRUE, useDemographicsAgeGroup = TRUE, useDemographicsRace
    = TRUE, useDemographicsEthnicity = TRUE, useDemographicsIndexYear = TRUE,
    useDemographicsIndexMonth = TRUE, useDemographicsTimeInCohort = TRUE,
  useDemographicsPriorObservationTime = TRUE, useDemographicsPostObservationTime =
   TRUE, useConditionGroupEraLongTerm = TRUE, useDrugGroupEraOverlapping = TRUE,
```

```
    useDrugGroupEraLongTerm = TRUE, useProcedureOccurrenceLongTerm = TRUE,

    useMeasurementLongTerm = TRUE, useObservationLongTerm = TRUE,
    useDeviceExposureLongTerm = TRUE, useVisitConceptCountLongTerm = TRUE,
    useConditionGroupEraShortTerm = TRUE, useDrugGroupEraShortTerm = TRUE,
    useProcedureOccurrenceShortTerm = TRUE, useMeasurementShortTerm = TRUE,
    useObservationShortTerm = TRUE, useDeviceExposureShortTerm = TRUE,
    useVisitConceptCountShortTerm = TRUE, useCharlsonIndex = TRUE, endDays = 0,
    longTermStartDays = -365, shortTermStartDays = -30)
)
```

Arguments

studyPopulationSettings	An object created using createStudyPopulationSettings or a list of createStudyPopulationSettin that specifies specific populations of interest
covariateSettings	An object created using FeatureExtraction::createCovariateSettings

Value

A list with the settings

See Also

Other Aggregate: createCaseSeriesSettings(), createRiskFactorSettings()

Examples

```
aggregateSetting <- createTargetBaselineSettings(
  studyPopulationSettings = createStudyPopulationSettings(
  targetIds = 1:2,
  limitToFirstInNDays = 99999,
  minPriorObservation = 365
  )
)
```

---

createTimeToEventSettings

Create time to event study settings

Description

Create time to event study settings

Usage

```
createTimeToEventSettings(studyPopulationSettings, outcomeIds)
```

Arguments

studyPopulationSettings	An object created using createStudyPopulationSettings or a list of createStudyPopulationSettin that specifies cohort inclusion criteria
outcomeIds	A list of cohortIds for the outcome cohorts

Value

An list with the time to event settings

Examples

```
# example code

tteSet <- createTimeToEventSettings(
  studyPopulationSettings = createStudyPopulationSettings(
    targetIds = c(1,2),
    limitToFirstInNDays = 0,
    minPriorObservation = 0
    ),
  outcomeIds = 3
)
```

---

exampleOmopConnectionDetails

create a connection detail for an example GI Bleed dataset from Eunomia

---

Description

This returns an object of class 'ConnectionDetails' that lets you connect via 'DatabaseConnector::connect()' to the example database.

Usage

```
exampleOmopConnectionDetails(exdir = tempdir())
```

Arguments

exdir	a directory to unzip the example OMOP database into. Default is tempdir().

Details

Finds the location of the example database in the package and calls 'DatabaseConnector::createConnectionDetails' to create a 'ConnectionDetails' object for connecting to the database.

Value

An object of class 'ConnectionDetails' with the details to connect to the example OHDSI OMOP CDM database

See Also

Other helper: `createStudyPopulationSettings()`

Examples

```
conDet <- exampleOmopConnectionDetails()

connectionHandler <- ResultModelManager::ConnectionHandler$new(conDet)
```

---

getDbDuringCovariateData

Extracts covariates that occur during a cohort

Description

Extracts covariates that occur during a cohort

Usage

```
getDbDuringCovariateData(
  connection,
  oracleTempSchema = NULL,
  cdmDatabaseSchema,
  cdmVersion = "5",
  cohortTable = "#cohort_person",
  rowIdField = "subject_id",
  aggregated = TRUE,
  cohortIds = c(-1),
  covariateSettings,
  minCharacterizationMean = 0,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
  targetDatabaseSchema = NULL,
  targetCovariateTable = NULL,
  targetCovariateContinuousTable = NULL,
  targetCovariateRefTable = NULL,
  targetAnalysisRefTable = NULL,
  targetTimeRefTable = NULL,
  ...
)
```

Arguments

connection	The database connection
oracleTempSchema	The temp schema if using oracle
cdmDatabaseSchema	The schema of the OMOP CDM data
cdmVersion	version of the OMOP CDM data
cohortTable	the table name that contains the target population cohort
rowIdField	string representing the unique identifier in the target population cohort
aggregated	whether the covariate should be aggregated
cohortIds	cohort id for the target cohort
covariateSettings	settings for the covariate cohorts and time periods
minCharacterizationMean	The minimum mean value for binary characterization output. Values below this will be cut off from output. This will help reduce the file size of the characterization output, but will remove information on covariates that have very low values. The default is 0.
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
targetDatabaseSchema	(Optional) The schema to save the tables targetCovariateTable/targetCovariateContinuousTable/targetCov when they are not temp tables and the output is being exported to database tables.
targetCovariateTable	(Optional) The name of the table where the resulting binary covariates will be stored. If not provided, results will be fetched to R. The table can be a permanent table in the `targetDatabaseSchema` or a temp table. If it is a temp table, do not specify `targetDatabaseSchema`.
targetCovariateContinuousTable	(Optional) The name of the table where the resulting continuous covariates will be stored. If not provided, results will be fetched to R. The table can be a permanent table in the `targetDatabaseSchema` or a temp table. If it is a temp table, do not specify `targetDatabaseSchema`.
targetCovariateRefTable	(Optional) The name of the table where the covariate reference will be stored.
targetAnalysisRefTable	(Optional) The name of the table where the analysis reference will be stored.
targetTimeRefTable	(Optional) The name of the table for the time reference
...	additional arguments from FeatureExtraction

Details

The user specifies a what during covariates they want and this executes them using FE

Value

A 'FeatureExtraction' covariateData object containing the during covariates based on user settings

See Also

Other CovariateSetting: createDuringCovariateSettings()

Examples

```
conDet <- exampleOmopConnectionDetails()
connection <- DatabaseConnector::connect(conDet)

settings <- createDuringCovariateSettings(
  useConditionOccurrenceDuring = TRUE,
  useConditionOccurrencePrimaryInpatientDuring = FALSE,
  useConditionEraDuring = FALSE,
  useConditionGroupEraDuring = FALSE
)

duringData <- getDbDuringCovariateData(
  connection <- connection,
  cdmDatabaseSchema = 'main',
  cohortIds = 1,
  covariateSettings = settings,
  cohortTable = 'cohort'
)

DatabaseConnector::disconnect(connection)
```

---

insertResultsToDatabase

Upload the results into a result database

Description

This function uploads results in csv format into a result database

Usage

```
insertResultsToDatabase(
  connectionDetails,
  schema,
  resultsFolder,
  tablePrefix = "",
```

```
  csvTablePrefix = "c_",
  includedFiles = NULL
)
```

Arguments

connectionDetails	The connection details to the result database
schema	The schema for the result database
resultsFolder	The folder containing the csv results
tablePrefix	A prefix to append to the result tables for the characterization results
csvTablePrefix	The prefix added to the csv results - default is 'c_'
includedFiles	Specify the csv files to upload or NULL to upload all in directory

Details

Calls ResultModelManager uploadResults function to upload the csv files

Value

Returns the connection to the sqlite database

See Also

Other Database: createCharacterizationTables(), createSqliteDatabase()

Examples

```
## generate results into resultsFolder
#conDet <- exampleOmopConnectionDetails()

#tteSet <- createTimeToEventSettings(
#  studyPopulationSettings = createStudyPopulationSettings(
#    targetIds = c(1,2),
#    limitToFirstInNDays = 0,
#    minPriorObservation = 0
#    ),
#  outcomeIds = 3
#  )

#cSet <- createCharacterizationSettings(
#  timeToEventSettings = tteSet
#)

#runCharacterizationAnalyses(
#  connectionDetails = conDet,
#  targetDatabaseSchema = 'main',
#  targetTable = 'cohort',
#  outcomeDatabaseSchema = 'main',
#  outcomeTable = 'cohort',
```

```
#  cdmDatabaseSchema = 'main',
#  characterizationSettings = cSet,
#  outputDirectory = file.path(tempdir(),'database')
#)

## create sqlite database
#charResultDbCD <- createSqliteDatabase()

## create database results tables
#createCharacterizationTables(
#   connectionDetails = charResultDbCD,
#   resultSchema = 'main'
# )

## insert results
#insertResultsToDatabase(
# connectionDetails = charResultDbCD,
# schema = 'main',
# resultsFolder = file.path(tempdir(),'database'),
# includedFiles = c('time_to_event')
#)
```

`loadCharacterizationSettings` *Load the characterization settings previously saved as a json file*

Description

This function converts the json file back into an R object

Usage

```
loadCharacterizationSettings(fileName)
```

Arguments

`fileName`	The location of the the json settings

Details

Input the directory containing the 'characterizationSettings.json' file and load the settings into R

Value

Returns the json settings as an R object

See Also

Other LargeScale: createCharacterizationSettings(), runCharacterizationAnalyses(), saveCharacterizationSettings()

Examples

```
# example code

setPath <- file.path(tempdir(), 'charSet.json')

drSet <- createDechallengeRechallengeSettings(
  studyPopulationSettings = createStudyPopulationSettings(
    targetIds = c(1,2),
    limitToFirstInNDays = 0,
    minPriorObservation = 0
    ),
  outcomeIds = 3
)

cSet <- createCharacterizationSettings(
  dechallengeRechallengeSettings = drSet
)

saveCharacterizationSettings(
  settings = cSet,
  fileName = setPath
)

setting <- loadCharacterizationSettings(setPath)
```

---

runCharacterizationAnalyses

execute a large-scale characterization study

Description

Specify the database connection containing the CDM data, the cohort database schemas/tables, the characterization settings and the directory to save the results to

Usage

```
runCharacterizationAnalyses(
  connectionDetails,
  targetDatabaseSchema,
  targetTable,
  outcomeDatabaseSchema,
  outcomeTable,
```

```
  nestingCohortTable = targetTable,
  nestingCohortDatabaseSchema = targetDatabaseSchema,
  outputDatabaseSchema = targetDatabaseSchema,
  outputTable = "characterization_cohort",
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
  cdmDatabaseSchema,
  characterizationSettings,
  outputDirectory,
  executionPath = file.path(outputDirectory, "execution"),
  csvFilePrefix = "c_",
  databaseId = "1",
  showSubjectId = FALSE,
  minCellCount = 0,
  incremental = TRUE,
  threads = 1,
  cohortGenerationThreads = NULL,
  nTargetJobs = 1,
  minCharacterizationMean = 0.001,
  minCovariateCount = 0,
  mode = "CohortIncidence",
  minSMD = 0,
  minTargetSize = 0,
  minCaseSize = 0
)
```

Arguments

connectionDetails	The connection details to the database containing the OMOP CDM data
targetDatabaseSchema	Schema name where your target cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
targetTable	Name of the target cohort table.
outcomeDatabaseSchema	Schema name where your outcome cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
outcomeTable	Name of the outcome cohort table.
nestingCohortTable	The cohort table to extract the nesting cohort from
nestingCohortDatabaseSchema	The schema containing the nestingCohortTable
outputDatabaseSchema	The schema where the characterization cohort table will be saved into
outputTable	The table name where the characterization cohort table will be saved into

`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

`cdmDatabaseSchema`
: The schema with the OMOP CDM data

`characterizationSettings`
: The study settings created using `createCharacterizationSettings`

`outputDirectory`
: The location to save the final csv files to

`executionPath`
: The location where intermediate results are saved to

`csvFilePrefix`
: A string to append the csv files in the outputDirectory

`databaseId`
: The unique identifier for the cdm database

`showSubjectId`
: Whether to include subjectId of failed rechallenge case series or hide

`minCellCount`
: The minimum count value that is calculated

`incremental`
: If TRUE then skip previously executed analyses that completed

`threads`
: The number of threads to use when running analyses jobs in parallel

`cohortGenerationThreads`
: (optional) The number of threads to use when generating the cohorts in parallel (Note: some database management systems do not allow insert parallelization)

`nTargetJobs`
: Partition the targets into this number of groups (e.g., if there are 20 targets and njobs is 5 then there will be 4 targets per job and 5 jobs)

`minCharacterizationMean`
: The minimum mean threshold to extract when running aggregate covariates

`minCovariateCount`
: The minimum number of patients who must have the covariate when running aggregate covariates

`mode`
: Select from Efficient (no exclusions to target based on washout)/CohortIncidence (excludes targets with outcome in washout if they have no time at risk)/PatientLevelPrediction (excludes targets with outcome during washout prior to index)

`minSMD`
: The minimum standardized mean difference for the risk factor analysis

`minTargetSize`
: The minimum target size to be included in targetBaseline, riskFactor or caseSeries

`minCaseSize`
: The minimum case or non-case size to be included in riskFactor or caseSeries

Details

The results of the characterization will be saved into an sqlite database inside the specified saveDirectory

Value

Multiple csv files in the outputDirectory.

See Also

Other LargeScale: createCharacterizationSettings(), loadCharacterizationSettings(), saveCharacterizationSettings()

Examples

```
conDet <- exampleOmopConnectionDetails()

tteSet <- createTimeToEventSettings(
  studyPopulationSettings = createStudyPopulationSettings(
    targetIds = c(1,2),
    limitToFirstInNDays = 0,
    minPriorObservation = 0
    ),
  outcomeIds = 3
)

cSet <- createCharacterizationSettings(
  timeToEventSettings = tteSet
)

runCharacterizationAnalyses(
  connectionDetails = conDet,
  targetDatabaseSchema = 'main',
  targetTable = 'cohort',
  outcomeDatabaseSchema = 'main',
  outcomeTable = 'cohort',
  cdmDatabaseSchema = 'main',
  characterizationSettings = cSet,
  outputDirectory = file.path(tempdir(),'runChar')
)
```

---

saveCharacterizationSettings

Save the characterization settings as a json

Description

This function converts the settings into a json object and saves it

Usage

```
saveCharacterizationSettings(settings, fileName)
```

Arguments

settings	An object of class characterizationSettings created using createCharacterizationSettings
fileName	The location to save the json settings

Details

Input the characterization settings and output a json file to a file named 'characterizationSettings.json' inside the saveDirectory

Value

Returns the location of the directory containing the json settings

See Also

Other LargeScale: createCharacterizationSettings(), loadCharacterizationSettings(), runCharacterizationAnalyses()

Examples

```
drSet <- createDechallengeRechallengeSettings(
  studyPopulationSettings = createStudyPopulationSettings(
    targetIds = c(1,2),
    limitToFirstInNDays = 0,
    minPriorObservation = 0
    ),
  outcomeIds = 3
)

cSet <- createCharacterizationSettings(
  dechallengeRechallengeSettings = drSet
)

saveCharacterizationSettings(
  settings = cSet,
  fileName = file.path(tempdir(), 'cSet.json')
)
```

`viewCharacterization` *viewCharacterization - Interactively view the characterization results*

Description

This is a shiny app for viewing interactive plots and tables

Usage

```
viewCharacterization(resultFolder, cohortDefinitionSet = NULL)
```

Arguments

`resultFolder`	The location of the csv results
`cohortDefinitionSet`	The cohortDefinitionSet extracted using webAPI

Details

Input is the output of ...

Value

Opens a shiny app for interactively viewing the results

Examples

```
conDet <- exampleOmopConnectionDetails()

tteSet <- createTimeToEventSettings(
studyPopulationSettings = createStudyPopulationSettings(
  targetIds = c(1,2)
  ),
  outcomeIds = 3
)

cSet <- createCharacterizationSettings(
  timeToEventSettings = tteSet
)

runCharacterizationAnalyses(
  connectionDetails = conDet,
  targetDatabaseSchema = 'main',
  targetTable = 'cohort',
  outcomeDatabaseSchema = 'main',
  outcomeTable = 'cohort',
  cdmDatabaseSchema = 'main',
  characterizationSettings = cSet,
  outputDirectory = file.path(tempdir(),'view')
)

# interactive shiny app
## Not run:
viewCharacterization(
  resultFolder = file.path(tempdir(),'view')
)

## End(Not run)
```

Index

∗ **Aggregate**
createCaseSeriesSettings, 4
createRiskFactorSettings, 13
createTargetBaselineSettings, 16
∗ **CohortIncidence**
createCohortIncidenceSettings, 8
∗ **CovariateSetting**
createDuringCovariateSettings, 10
getDbDuringCovariateData, 19
∗ **Database**
createCharacterizationTables, 7
createSqliteDatabase, 14
insertResultsToDatabase, 21
∗ **DechallengeRechallenge**
createDechallengeRechallengeSettings, 9
∗ **Incremental**
cleanIncremental, 2
cleanNonIncremental, 3
∗ **LargeScale**
createCharacterizationSettings, 5
loadCharacterizationSettings, 23
runCharacterizationAnalyses, 24
saveCharacterizationSettings, 27
∗ **Shiny**
viewCharacterization, 28
∗ **TimeToEvent**
createTimeToEventSettings, 17
∗ **helper**
createStudyPopulationSettings, 15
exampleOmopConnectionDetails, 18

cleanIncremental, 2
cleanIncremental(), *3*
cleanNonIncremental, 3
cleanNonIncremental(), *3*
createCaseSeriesSettings, 4
createCaseSeriesSettings(), *14*, *17*
createCharacterizationSettings, 5
createCharacterizationSettings(), *24*, *27*, *28*
createCharacterizationTables, 7
createCharacterizationTables(), *15*, *22*
createCohortIncidenceSettings, 8
createDechallengeRechallengeSettings, 9
createDuringCovariateSettings, 10
createDuringCovariateSettings(), *21*
createRiskFactorSettings, 13
createRiskFactorSettings(), *5*, *17*
createSqliteDatabase, 14
createSqliteDatabase(), *7*, *22*
createStudyPopulationSettings, 15
createStudyPopulationSettings(), *19*
createTargetBaselineSettings, 16
createTargetBaselineSettings(), *5*, *14*
createTimeToEventSettings, 17

exampleOmopConnectionDetails, 18
exampleOmopConnectionDetails(), *16*

getDbDuringCovariateData, 19
getDbDuringCovariateData(), *12*

insertResultsToDatabase, 21
insertResultsToDatabase(), *7*, *15*

loadCharacterizationSettings, 23
loadCharacterizationSettings(), *6*, *27*, *28*

runCharacterizationAnalyses, 24
runCharacterizationAnalyses(), *6*, *24*, *28*

saveCharacterizationSettings, 27
saveCharacterizationSettings(), *6*, *24*, *27*

viewCharacterization, 28