Package 'Capr'

February 18, 2025

Title Cohort Definition Application Programming

Version 2.0.9

Description Provides a programming language for defining cohort definitions in R to use in studies for Observational
Health Data Sciences and Informatics (OHDSI). The functions in 'Capr' allow for the programmatic creation of
OHDSI concept sets and cohorts that can be serialized to 'OHDSI' compatible 'json' files or to 'OHDSI-SQL'.
'Capr' functions can be used to create, save, and load component parts to a cohort definition allowing
R programmers to easily reuse cohort logic. 'Capr' provides tools to create a large number of OHDSI cohorts
while also helping bridge the gap between human readable descriptions of clinical phenotypes and their computational implementation.

License Apache License (>= 2)

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

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

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 3.5.0)

Imports magrittr (>= 1.5.0),
jsonlite,
methods,
purrr (>= 1.0.1),
rlang,
dplyr,
tidyr,
checkmate,
tibble,
readr,
stringr,
glue,
cli,
digest,
fs,
lubridate,
DBI,

DatabaseConnector,
SqlRender,
generics,
CirceR

Suggests testthat (>= 3.0.0),
knitr,
rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

Collate 'Capr.R'
'conceptSet.R'
'attributes-concept.R'
'attributes-dateAdjustment.R'
'attributes-logic.R'
'query.R'
'window.R'
'criteria.R'
'attributes-nested.R'
'attributes-op.R'
'exit.R'
'cohort.R'
'collectCodesetId.R'
'utils.R'

R topics documented:

`==` *FUnction checks if two concept set class objects are equivalent*

Description

FUnction checks if two concept set class objects are equivalent

Usage

```
## S4 method for signature 'ConceptSet,ConceptSet'
e1 == e2
```

Arguments

`e1, e2`	a ConceptSet Class object

`age` *Function to create age attribute*

Description

Function to create age attribute

Usage

```
age(op)
```

Arguments

op	an opAttribute object that is either numeric or integer that defines the logical operation used to determine eligible patient age

Value

An age attribute that can be used in a cohort definition

---

as.data.frame,ConceptSet-method *Coerce a concept set expression to a dataframe*

Description

Coerce a concept set expression to a dataframe

Usage

```
## S4 method for signature 'ConceptSet'
as.data.frame(x)
```

Arguments

x	A Caper Concept Set

Value

A tibble (dataframe) with columns: concept_id, includeDescendants, isExcluded, includeMapped.

---

as.json *Coerce Capr object to json*

Description

Coerce Capr object to json

Usage

```
as.json(x, pretty = TRUE, ...)

## S4 method for signature 'ConceptSet'
as.json(x, pretty = TRUE, ...)

## S4 method for signature 'Cohort'
as.json(x, pretty = TRUE, ...)
```

Arguments

x	the capr object
pretty	a toggle to make the json look nice, part of jsonlite
...	additional arguments passes to jsonlite::toJSON

atLeast *Function to enumerate an minimal count of occurrences*

Description

Function to enumerate an minimal count of occurrences

Usage

```
atLeast(x, query, aperture = duringInterval(eventStarts(-Inf, Inf)))
```

Arguments

x	the integer counting the number of occurrences
query	a query object that provides context to the clinical event of interest
aperture	an eventAperture object that shows the temporal span where the event is to be observed relative to the index event

Value

A criteria that can be used in a cohort definition specifying satisfaction of at least x instances of a query

atMost *Function to enumerate a maximum count of occurrences*

Description

Function to enumerate a maximum count of occurrences

Usage

```
atMost(x, query, aperture = duringInterval(eventStarts(-Inf, Inf)))
```

Arguments

x	the integer counting the number of occurrences
query	a query object that provides context to the clinical event of interest
aperture	an eventAperture object that shows the temporal span where the event is to be observed relative to the index event

Value

A criteria that can be used in a cohort definition specifying satisfaction of at most x instances of a query

attrition *Create a cohort attrition object*

Description

Create a cohort attrition object

Usage

```
attrition(..., expressionLimit = c("First", "All", "Last"))
```

Arguments

`...`	Capr groups
`expressionLimit`	how to limit initial events per person either First, All, or Last

Value

A cohort attrition object that can be used in a cohort definition

bt *Between operator*

Description

function that builds an opAttribute based on between logic

Usage

```
bt(x, y)

## S4 method for signature 'integer'
bt(x, y)

## S4 method for signature 'numeric'
bt(x, y)

## S4 method for signature 'Date'
bt(x, y)
```

Arguments

`x`	the left side bound of the between logic This can either be an integer, numeric, or Date data type. Different data types will return the appropriate opAttribute type
`y`	the right side bound of the between logic. This can either be an integer, numeric, or Date data type. Different data types will return the appropriate opAttribute type

CensoringCriteria-class *An S4 class identifying a censoring criteria for the cohort*

Description

The censoring criteria specifies events where the person exits the cohort. These events are based on a query class object and users can specify multiple queries in the censoring criteria.

Slots

`criteria` a list of Capr query class objects that specify the events that would lead a person to exit the cohort.

censoringEvents *Constructor for a set of censoring events*

Description

Constructor for a set of censoring events

Usage

```
censoringEvents(...)
```

Arguments

`...`	a list of Capr query objects that are used as censoring events

Value

a censoring criteria S4 object used to define the censoring events of the cohort definition

cohort *Function that creates a cohort object*

Description

Function that creates a cohort object

Usage

```
cohort(entry, attrition = NULL, exit = NULL, era = NULL)
```

Arguments

entry	the index event of the cohort
attrition	rules that restrict the cohort further, developing attrition
exit	the event where the person exits the cohort
era	Cohort era (collapse) logic created with the ‘cohortEra‘ function

Value

an S4 Cohort class object describing the cohort definiton

---

compile *Compile a Capr object to json*

Description

Compile a Capr object to json

Usage

```
compile(object, ...)
```

Arguments

object	A Capr object such as a cohort, list of cohorts, or concept set.
...	Arguments passed on to jsonlite::toJSON. e.g. ‘pretty = TRUE‘ for nicely formatted json.

Value

The json representation of the Capr object

---

compile,Cohort-method *Compile a Capr cohort to json*

Description

Compile a Capr cohort to json

Compile a Capr Concept Set to json

Usage

```
## S4 method for signature 'Cohort'
compile(object, ...)

## S4 method for signature 'ConceptSet'
compile(object, ...)
```

Arguments

object	A Capr Concept Set created with ‘cs‘
...	Arguments passed on to jsonlite::toJSON. e.g. ‘pretty = TRUE‘ for nicely formatted json.

Value

The json representation of Capr cohorts

The json representation of Capr cohorts

Examples

```
ch <- cohort(conditionOccurrence(cs(1,2, name = "concepts")))
compile(ch)
```

Concept-class *An S4 class for a single OMOP Concept*

Description

A concept class contains all the information about the concept from the OMOP voabulary.

Slots

`concept_id` the OMOP/OHDSI concept ID

`concept_name` the name of the concept

`standard_concept` whether the concept is standard ’S’, classification ’C’, or non-standard NA

`standard_concept_caption` Whether the concept is standard full phrase

`invalid_reason` Whether the concept is invalid single letter

`invalid_reason_caption` whether the concept is invalid standard phrase

`concept_code` The original code of the concept from its vocabulary

`domain_id` The domain of the concept (e.g. Drug, Condition, Procedure, etc)

`vocabulary_id` the name of the vocabulary

`concept_class_id` type of concept class

---

conceptAttribute-class

An S4 class for a concept attribute

Description

An S4 class for a concept attribute

Slots

`name` the name of the attribute

`conceptSet` a list representing the concepts for the attribute

ConceptSet-class *An S4 class for ConceptSetExpresion*

Description

A class for the concept set expressions bundles multiple concepts with mapping

Slots

`id` an id for the concept set expression to identify within a component

`Name` the name of the concept set expression

`Expression` a list containing expressions. expressions include multiple conceptSetItem objects

ConceptSetItem-class *An S4 class for ConceptSetItem*

Description

A class that provides information on the mapping of the concept

Slots

`Concept` a concept class object

`isExcluded` toggle if want to exclude the concept

`includeDescendants` toggle if want to include descendants

`includeMapped` toggle if want to include map

conditionEra *Query the condition era domain*

Description

Query the condition era domain

Usage

```
conditionEra(conceptSet, ...)
```

Arguments

`conceptSet`	A condition concept set
`...`	optional attributes

Value

A Capr Query

conditionOccurrence *Query the condition domain*

Description

Query the condition domain

Usage

```
conditionOccurrence(conceptSet, ...)
```

Arguments

conceptSet	A condition concept set
...	optional attributes

Value

A Capr Query

conditionType *Add a condition type attribute to determine the provenance of the record*

Description

Add a condition type attribute to determine the provenance of the record

Usage

```
conditionType(ids, connection, vocabularyDatabaseSchema)
```

Arguments

ids	the concept ids for the attribute
connection	a connection to an OMOP dbms to get vocab info about the concept
vocabularyDatabaseSchema	the database schema for the vocabularies

Value

An attribute that can be used in a query function

continuousObservation *A function to construct the observationWindow*

Description

A function to construct the observationWindow

Usage

```
continuousObservation(priorDays = 0L, postDays = 0L)
```

Arguments

priorDays	minimum number of observation days prior to the cohort index. Default 0 days
postDays	minimum number of observation days post cohort index. Default 0 days

Value

An observation window that can be used in a Capr cohort definition

Criteria-class *An S4 for a criteria*

Description

a criteria is a temporal observation of a clinical event relative to the index event

Slots

occurrence an occurrence object specifying how many events must occur to consider the event as part of the cohort definition

query a query object that provides context to the clinical event of interest

aperture an eventAperture object that shows the temporal span where the event is to be observed relative to the index event

cs *Create a concept set*

Description

cs is used to create concept set expressions.

'exclude' is meant to be used inside 'cs' when creating a new concept set.

'mapped' is meant to be used inside 'cs' when creating a new concept set.

'descendants' is meant to be used inside 'cs' when creating a new concept set.

Usage

```
cs(..., name, id = NULL)

exclude(...)

mapped(...)

descendants(...)
```

Arguments

...	One or more numeric vectors that can be coerced to integers, or Calls to helper functions "exclude", "descendants", or "mapped".
name	A name for the concept set
id	An id for the concept set

Value

A Capr Concept Set Object

A list of Capr concepts

A list of Capr concepts

A list of Capr concepts

Functions

- `exclude()`: exclude concepts
- `mapped()`: Include mapped concepts
- `descendants()`: Include descendants

Examples

```
cs(1, 2, name = "concepts")
cs(1, c(10, 11, 2), name = "concepts")
cs(1, seq(2, 10, 2), name = "concepts")
cs(1, 2, 3, exclude(4, 5), name = "concepts")
cs(1, 2, 3, exclude(4, 5), mapped(6, 7), name = "concepts")
cs(1, 2, 3, exclude(4, 5), mapped(6, 7), descendants(8, 9), name = "concepts")
cs(descendants(1, 2, 3),  exclude(descendants(8, 9)), name = "concepts")
```

dateAdjustment *Function to create age attribute*

Description

Function to create age attribute

Usage

```
dateAdjustment(
  startWith = "START_DATE",
  startOffset = 0L,
  endWith = "END_DATE",
  endOffset = 0L
)
```

Arguments

startWith	character string either START_DATE or END_DATE
startOffset	an integer value, default 0
endWith	character string either START_DATE or END_DATE
endOffset	an integer value, default 0

Value

A dateAdjustment attribute class that can be used with a query

dateAdjustmentAttribute-class *An S4 class for a date adjustment attribute*

Description

An S4 class for a date adjustment attribute

Slots

name the name of the attribute

startWith character string either START_DATE or END_DATE

startOffset an integer value, default 0

endWith character string either START_DATE or END_DATE

endOffset an integer value, default 0

daysOfSupply *Function to create days supply attribute*

Description

This function is used only for a drug query. days supply is a column in the drug exposure table of the cdm. This attribute allows a subquery to find drugs that satisfy certain values determined by the op logic.

Usage

```
daysOfSupply(op)
```

Arguments

op	an opAttribute object that is either numeric or integer that defines the logical operation used to determine eligible number of days of supply

Value

An attribute that can be used in a cohort definition

death *Query the condition era domain*

Description

Query the condition era domain

Usage

```
death(conceptSet = NULL, ...)
```

Arguments

conceptSet	A condition concept set
...	optional attributes

Value

A Capr Query

deviceExposure *Query the drug domain*

Description

Query the drug domain

Usage

```
deviceExposure(conceptSet, ...)
```

Arguments

conceptSet	A drug concept set
...	optional attributes

Value

A Capr Query

drugEra *Query the drug era domain*

Description

Query the drug era domain

Usage

```
drugEra(conceptSet, ...)
```

Arguments

conceptSet	A drug ingredient concept set
...	optional attributes

Value

A Capr Query

drugExit — *Function to create an exit based on exit based on the end of a continuous drug exposure*

Description

Function to create an exit based on exit based on the end of a continuous drug exposure

Usage

```
drugExit(
  conceptSet,
  persistenceWindow = 0L,
  surveillanceWindow = 0L,
  daysSupplyOverride = NULL
)
```

Arguments

conceptSet	the concept set of the drug exposure used to identify the exit
persistenceWindow	allow for a maximum of days between exposure records when inferring the era of persistence exposure
surveillanceWindow	add days to the end of the era of persistence exposure as an additional period of surveillance prior to cohort exit
daysSupplyOverride	force drug exposure days supply to a set number of days

Value

an S4 DrugExposueExit class which defines the cohort exit by end of drug exposure

drugExposure — *Query the drug domain*

Description

Query the drug domain

Usage

```
drugExposure(conceptSet, ...)
```

Arguments

conceptSet	A drug concept set
...	optional attributes

Value

A Capr Query

`DrugExposureExit-class` *An S4 class for a cohort exit based on continuous exposure persistence.*

Description

Specify a concept set that contains one or more drugs. A drug era will be derived from all drug exposure events for any of the drugs within the specified concept set, using the specified persistence window as a maximum allowable gap in days between successive exposure events and adding a specified surveillance window to the final exposure event. If no exposure event end date is provided, then an exposure event end date is inferred to be event start date + days supply in cases when days supply is available or event start date + 1 day otherwise. This event persistence assures that the cohort end date will be no greater than the drug era end date.

Slots

`conceptSet` the concept set of the drug exposure used to identify the exit

`persistenceWindow` allow for a maximum of days between exposure records when inferring the era of persistence exposure

`surveillanceWindow` add days to the end of the era of persistence exposure as an additional period of surveillance prior to cohort exit

`daysSupplyOverride` force drug exposure days supply to a set number of days

`count` an integer specifying the number of occurrences for a criteria

---

`drugQuantity` *Function to create quantity attribute*

Description

This function is used only for a drug query. quantity is a column in the drug exposure table of the cdm. This attribute allows a subquery to find drugs that satisfy certain values determined by the op logic.

Usage

```
drugQuantity(op)
```

Arguments

`op`	an opAttribute object that is either numeric or integer that defines the logical operation used to determine eligible quantity

Value

An attribute that can be used in a cohort definition

drugRefills *Function to create refills attribute*

Description

This function is used only for a drug query. refills is a column in the drug exposure table of the cdm. This attribute allows a subquery to find drugs that satisfy certain values determined by the op logic.

Usage

```
drugRefills(op)
```

Arguments

op	an opAttribute object that is either numeric or integer that defines the logical operation used to determine eligible number of refills

Value

An attribute that can be used in a cohort definition

drugType *Add a drug type attribute to determine the provenance of the record*

Description

Add a drug type attribute to determine the provenance of the record

Usage

```
drugType(ids, connection, vocabularyDatabaseSchema)
```

Arguments

ids	the concept ids for the attribute
connection	a connection to an OMOP dbms to get vocab info about the concept
vocabularyDatabaseSchema	the database schema for the vocabularies

Value

An attribute that can be used in a query function

duringInterval *Function that creates an eventAperture an opening where an event can occur relative to the index event*

Description

Function that creates an eventAperture an opening where an event can occur relative to the index event

Usage

```
duringInterval(
  startWindow,
  endWindow = NULL,
  restrictVisit = FALSE,
  ignoreObservationPeriod = FALSE
)
```

Arguments

startWindow	the starting window where an event can occur
endWindow	the end window of where an event can occur. This parameter is optional
restrictVisit	a logical toggle specifying whether the event should occur on the same visit
ignoreObservationPeriod	a logical toggle specifying whether we can consider events outside the observation period

Value

An event aperture that can be used in a Capr cohort definition

endDate *Function that creates a end date attribute*

Description

Function that creates a end date attribute

Usage

```
endDate(op, type = "occurrence")
```

Arguments

op	an opAttribute object must be a date that defines the logical operation used to determine eligible end dates
type	specify the type of date to use either occurrence or era. default as occurrence

Value

An attribute that can be used in a cohort definition

Endpoint-class *An S4 class for an Endpoint*

Description

this determines the time in days relative to an index either before or after

Slots

`days` either a character string all or an integer for the number of days

`coeff` a character string either before or after

entry *Create a cohort entry criteria*

Description

Create a cohort entry criteria

Usage

```
entry(
  ...,
  observationWindow = continuousObservation(0L, 0L),
  primaryCriteriaLimit = c("First", "All", "Last"),
  additionalCriteria = NULL,
  qualifiedLimit = c("First", "All", "Last")
)
```

Arguments

`...`	Capr Queries
`observationWindow`	a time specifying the minimal time a person is observed
`primaryCriteriaLimit`	Which primary criteria matches should be considered for inclusion? "First", "Last" or "All"
`additionalCriteria`	a Capr group that adds restriction to the entry event
`qualifiedLimit`	Which criteria matches should be considered for inclusion? "First", "Last" or "All"

Value

A cohort entry Capr object

eq *Equal to operator*

Description

function that builds an opAttribute based on equal to logic

Usage

```
eq(x)

## S4 method for signature 'integer'
eq(x)

## S4 method for signature 'numeric'
eq(x)

## S4 method for signature 'Date'
eq(x)
```

Arguments

x	the value to used as a bound in the op logic. This can either be an integer, numeric, or Date data type. Different data types will return the appropriate opAttribute type

era *Create a Cohort Era class object*

Description

The Cohort Era depicts the time span of the cohort. The Censor Window includes the date window for which we register events. The Collapse Settings identify the era padding between events before exiting a cohort.

Usage

```
era(eraDays = 0L, studyStartDate = NULL, studyEndDate = NULL)
```

Arguments

eraDays	a numeric that specifies the number of days for the era padding
studyStartDate	a date string that specifies the starting date of registration
studyEndDate	a date string that specifies the end date of registration

Value

a S4 CohortEra class object defining the eras of the cohort definition

EventAperture-class *An S4 class for Aperture*

Description

The aperture class provides context to when the criteria must be observed in a person timeline to pretain to the expression

Slots

`startWindow` a EventWindow class object identifying the start window

`endWindow` a EventWindow class object ifentifying the end window (optional)

`restrictVisit` a logic toggle where TRUE restricts to the same visit

`ignoreObservationPeriod` a logic toggle where TRUE allows events outside the observation period

eventEnds *Function creates an event window where the event ends*

Description

Function creates an event window where the event ends

Usage

```
eventEnds(a, b, index = c("startDate", "endDate"))
```

Arguments

a	the left side of the event window
b	the right side of the event window
index	specifying what part of the index we start looking for events either at the index start date or index enddate

Value

An event window that can be used in a Capr cohort definition

eventStarts *Function creates an event window where the event starts*

Description

Function creates an event window where the event starts

Usage

```
eventStarts(a, b, index = c("startDate", "endDate"))
```

Arguments

a	the left side of the event window
b	the right side of the event window
index	specifying what part of the index we start looking for events either at the index start date or index enddate

Value

An event window that can be used in a Capr cohort definition

EventWindow-class *An S4 class for a Window*

Description

A window class provides details on the end points of the timeline

Slots

`event` a character string either start or end. Identifies the point of reference for the window

`start` an endpoint object containing the days and coefficient for the start of the window

`end` an endpoint object containing the days and coefficient for the end of the window

`index` A character string either start or end. Identifies where the index is relative to the window

exactly *Function to enumerate an exact count of occurrences*

Description

Function to enumerate an exact count of occurrences

Usage

```
exactly(x, query, aperture = duringInterval(eventStarts(-Inf, Inf)))
```

Arguments

x	the integer counting the number of occurrences
query	a query object that provides context to the clinical event of interest
aperture	an eventAperture object that shows the temporal span where the event is to be observed relative to the index event

Value

A criteria that can be used in a cohort definition specifying satisfaction of exactly x instances of a query

exit *Function that creates a cohort exit object*

Description

Function that creates a cohort exit object

Usage

```
exit(endStrategy, censor = NULL)
```

Arguments

endStrategy	the endStrategy object to specify for the exit
censor	the censoring criteria to specify for the exit

Value

A cohort exit object that can be used in a cohort definition

`firstOccurrence` *Add first occurrence attribute*

Description

Add first occurrence attribute

Usage

```
firstOccurrence()
```

Value

An attribute that can be used in a query function

`FixedDurationExit-class` *An S4 class for a cohort exit based on fixed duration persistence.*

Description

The event end date is derived from adding a number of days to the event's start or end date. If an offset is added to the event's start date, all cohort episodes will have the same fixed duration (subject to further censoring). If an offset is added to the event's end date, persons in the cohort may have varying cohort duration times due to the varying event durations (such as eras of persistent drug exposure or visit length of stay). This event persistence assures that the cohort end date will be no greater than the selected index event date, plus the days offset.

Slots

`index` specification of event date to offset

`offsetDays` an integer specifying the number of days to offset from the event date

`fixedExit` *Function to create an exit based on exit based on the end of a continuous drug exposure*

Description

Function to create an exit based on exit based on the end of a continuous drug exposure

Usage

```
fixedExit(index = c("startDate", "endDate"), offsetDays)
```

Arguments

index	specification of event date to offset. Can be either startDate or endDate
offsetDays	an number specifying the days to offset from the event date. Will coerce to an integer

Value

a fixed Duration exit S4 object used to define the cohort exit as the end of a specified time

---

getConceptSetCall *Create the Capr code to build a concept set*

Description

Create the Capr code to build a concept set

Usage

```
getConceptSetCall(x, name = x@Name)
```

Arguments

x	A concept set
name	the name of the concept set

Value

The Capr code required to build the concept set

---

getConceptSetDetails *Fill in Concept Set details using a vocab*

Description

Concept sets created in R using the ‘cs‘ function do not contain details like "CONCEPT_NAME", "DOMAIN_ID", etc. If an OMOP CDM vocabulary is available then these details can be filled in by the the ‘getConceptSetDetails‘ function.

Usage

```
getConceptSetDetails(x, con, vocabularyDatabaseSchema = NULL)
```

Arguments

x	A concept set created by ‘cs‘
con	A connection to an OMOP CDM database
vocabularyDatabaseSchema	Schema name where your OMOP vocabulary format resides. Note that for SQL Server, this should include both the database and schema name, for example ’vocabulary.dbo’.

Value

A modified version of the input concept set with concept details filled in.

Examples

```
## Not run:
# create a concept set
vocabularyDatabaseSchema = "cdm5"
anemia <- cs(descendants(439777,4013073,4013074), name = "anemia")

# fill in the details from an OMOP CDM
library(DatabaseConnector)
con <- connect(dbms = "postgresql", user = "postgres", password = "", server = "localhost/cdm")
anemia <- getConceptSetDetails(condition_anemia, con, vocabularyDatabaseSchema = "cdm5")

## End(Not run)
```

Group-class *An S4 class for a group*

Description

a group is the combination of multiple criteria or sub groups

Slots

`occurrence` an occurrence object specifying how many events must occur to consider the event as part of the cohort definition

`critera` a list of criteria that are grouped together

`group` a list of sub-groups to consider

---

gt *Greater than operator*

Description

function that builds an opAttribute based on greater than logic

Usage

```
gt(x)

## S4 method for signature 'integer'
gt(x)

## S4 method for signature 'numeric'
gt(x)

## S4 method for signature 'Date'
gt(x)
```

Arguments

x	the value to used as a bound in the op logic. This can either be an integer, numeric, or Date data type. Different data types will return the appropriate opAttribute type

---

gte *Greater than or equal to operator*

Description

function that builds an opAttribute based on greater than or equal to logic

Usage

```
gte(x)

## S4 method for signature 'integer'
gte(x)

## S4 method for signature 'numeric'
gte(x)

## S4 method for signature 'Date'
gte(x)
```

Arguments

x	the value to used as a bound in the op logic. This can either be an integer, numeric, or Date data type. Different data types will return the appropriate opAttribute type

---

logicAttribute-class *An S4 class for a logical attribute*

Description

with a logic attribute if it is specified than we assume it is true

Slots

name the name of the attribute

lt *Less than operator*

Description

function that builds an opAttribute based on less than logic

Usage

```
lt(x)

## S4 method for signature 'integer'
lt(x)

## S4 method for signature 'numeric'
lt(x)

## S4 method for signature 'Date'
lt(x)
```

Arguments

x	the value to used as a bound in the op logic. This can either be an integer, numeric, or Date data type. Different data types will return the appropriate opAttribute type

lte *Less than or equal to operator*

Description

function that builds an opAttribute based on less than or equal to than logic

Usage

```
lte(x)

## S4 method for signature 'integer'
lte(x)

## S4 method for signature 'numeric'
lte(x)

## S4 method for signature 'Date'
lte(x)
```

Arguments

x	the value to used as a bound in the op logic. This can either be an integer, numeric, or Date data type. Different data types will return the appropriate opAttribute type

makeCohortSet *Make a cohort dataframe for cohort generator*

Description

Make a cohort dataframe for cohort generator

Usage

```
makeCohortSet(...)
```

Arguments

...	multiple capr cohorts to bind into a dataframe

Value

a tibble containing cohortId, name, sql and json to pipe into CohortGenerator.

male *Add male attribute to a query*

Description

Add male attribute to a query

Add female attribute to a query

Usage

```
male()

female()
```

Value

An attribute that can be used in a query function

An attribute that can be used in a query function

Functions

- `male()`: male demographic attribute
- `female()`: female demographic attribute

Examples

```
# Create a cohort of males with Type 1 diabetes
t1dm <- cs(descendants(201254, 435216, 40484648), name = "type 1 diabetes")
t1dm_males <- cohort(conditionOccurrence(t1dm, male()))
# Create a cohort of males with Type 1 diabetes
t1dm <- cs(descendants(201254, 435216, 40484648), name = "type 1 diabetes")
t1dm_females <- cohort(conditionOccurrence(t1dm, female()))
```

measurement *Query the measurement domain*

Description

Query the measurement domain

Usage

```
measurement(conceptSet, ...)
```

Arguments

conceptSet	A measurement concept set
...	optional attributes

Value

A Capr Query

measurementType *Add a measurement type attribute to determine the provenance of the record*

Description

Add a measurement type attribute to determine the provenance of the record

Usage

```
measurementType(ids, connection, vocabularyDatabaseSchema)
```

Arguments

ids	the concept ids for the attribute
connection	a connection to an OMOP dbms to get vocab info about the concept
vocabularyDatabaseSchema	the database schema for the vocabularies

Value

An attribute that can be used in a query function

nbt *Not between operator*

Description

function that builds an opAttribute based on not between logic

Usage

```
nbt(x, y)

## S4 method for signature 'integer'
nbt(x, y)

## S4 method for signature 'numeric'
nbt(x, y)

## S4 method for signature 'Date'
nbt(x, y)
```

Arguments

x	the left side bound of the between logic This can either be an integer, numeric, or Date data type. Different data types will return the appropriate opAttribute type
y	the right side bound of the between logic. This can either be an integer, numeric, or Date data type. Different data types will return the appropriate opAttribute type

nestedAttribute-class *An S4 class for a nested attribute*

Description

An S4 class for a nested attribute

Slots

`name` the name of the attribute

`conceptSet` a list representing the concepts for the attribute

`nestedWithAll` *Construct a nested group where all criteria and groups must be satisfied*

Description

Construct a nested group where all criteria and groups must be satisfied

Usage

```
nestedWithAll(...)
```

Arguments

`...`	a set of criteria or groups

`nestedWithAny` *Function to construct a nested group where any criteria and groups may be satisfied*

Description

Function to construct a nested group where any criteria and groups may be satisfied

Usage

```
nestedWithAny(...)
```

Arguments

`...`	a set of criteria or groups

`nestedWithAtLeast` *Function to construct a nested group where at least some of the criteria or groups must be satisfied*

Description

Function to construct a nested group where at least some of the criteria or groups must be satisfied

Usage

```
nestedWithAtLeast(x, ...)
```

Arguments

`x`	an integer specifying the number of criteria or groups that must be satisfied
`...`	a set of criteria or groups

nestedWithAtMost — *Function to construct a nested group where at most some of the criteria or groups must be satisfied*

Description

Function to construct a nested group where at most some of the criteria or groups must be satisfied

Usage

```
nestedWithAtMost(x, ...)
```

Arguments

x	an integer specifying the number of criteria or groups that must be satisfied
...	a set of criteria or groups

observation — *Query the observation domain*

Description

Query the observation domain

Usage

```
observation(conceptSet, ...)
```

Arguments

conceptSet	A condition concept set
...	optional attributes

Value

A Capr Query

observationExit — *Function to create an exit based on continuous observation*

Description

Function to create an exit based on continuous observation

Usage

```
observationExit()
```

Value

an S4 ObservationExit class which defines the cohort exit as the end of continuous observation

ObservationExit-class *An S4 class for a cohort exit based on end of continuous observation.*

Description

The event persists until the end of continuous observation of the persons

Slots

`index` specification of event date to offset

`offsetDays` an integer specifying the number of days to offset from the event date

observationType *Add a observation type attribute to determine the provenance of the record*

Description

Add a observation type attribute to determine the provenance of the record

Usage

```
observationType(ids, connection, vocabularyDatabaseSchema)
```

Arguments

`ids`	the concept ids for the attribute
`connection`	a connection to an OMOP dbms to get vocab info about the concept
`vocabularyDatabaseSchema`	the database schema for the vocabularies

Value

An attribute that can be used in a query function

ObservationWindow-class *An S4 class for an ObservationWindow*

Description

this determines the minimal observation time before and after index for all persons in the cohort

Slots

`priorDays` minimum number of days prior to the cohort index

`postDays` minimum number of days post cohort index

Occurrence-class *An S4 class for an occurrence.*

Description

This determines how many events need to occur to count the criteria in the cohort definition (relative to the index event)

Slots

`type` a character string determine the logic for counting occurrences. Can be all, any, exactly, atLeast, or atMost

`count` an integer specifying the number of occurrences for a criteria

opAttributeDate-class *An S4 class for a op attribute that is a date*

Description

An S4 class for a op attribute that is a date

Slots

`name` the name of the attribute

`op` the operator one of: gt,lt,gte,lte,eq,bt,!bt

`value` a value serving as the single limit or lower limit in a bt.

`extent` a value serving as the upper limit in a bt, otherwise this is empty

opAttributeInteger-class *An S4 class for a op attribute that is an integer*

Description

An S4 class for a op attribute that is an integer

Slots

`name` the name of the attribute

`op` the operator one of: gt,lt,gte,lte,eq,bt,!bt

`value` a value serving as the single limit or lower limit in a bt.

`extent` a value serving as the upper limit in a bt, otherwise this is empty

opAttributeNumeric-class *An S4 class for a op attribute that is a numeric*

Description

An S4 class for a op attribute that is a numeric

Slots

`name` the name of the attribute

`op` the operator one of: gt,lt,gte,lte,eq,bt,!bt

`value` a value serving as the single limit or lower limit in a bt

`extent` a value serving as the upper limit in a bt, otherwise this is empty

opAttributeSuper-class *An S4 super class for other opAttribute objects to inherit.*

Description

An S4 super class for other opAttribute objects to inherit.

procedure *Query the procedure domain*

Description

Query the procedure domain

Usage

```
procedure(conceptSet, ...)
```

Arguments

`conceptSet`	A procedure concept set
`...`	optional attributes

Value

A Capr Query

procedureType — *Add a procedure type attribute to determine the provenance of the record*

Description

Add a procedure type attribute to determine the provenance of the record

Usage

```
procedureType(ids, connection, vocabularyDatabaseSchema)
```

Arguments

ids	the concept ids for the attribute
connection	a connection to an OMOP dbms to get vocab info about the concept
vocabularyDatabaseSchema	the database schema for the vocabularies

Value

An attribute that can be used in a query function

Query-class — *An S4 class for a Circe Query*

Description

A query is a medical concept that can be extracted from a database through a 'where' clause in a 'SQL' statement. This includes concepts.

Slots

domain	The domain to search (e.g. "Condition", "Drug", "Measurement", etc)
conceptSet	The Concept set describing the observation to serach for
attributes	a list of attributes that modify the query (e.g. 'male()', 'female()', 'age(gte(65))')

rangeHigh *Function to create rangeHigh attribute*

Description

This function is used only for measurement query. range_high is a column in the measurement table of the cdm. This attribute allows a subquery to find measurements that satisfy certain values determined by the op logic.

Usage

```
rangeHigh(op)
```

Arguments

op	an opAttribute object that is either numeric or integer that defines the logical operation used to determine eligible range high

Value

An attribute that can be used in a cohort definition

rangeLow *Function to create rangeLow attribute*

Description

This function is used only for measurement query. range_low is a column in the measurement table of the cdm. This attribute allows a subquery to find measurements that satisfy certain values determined by the op logic.

Usage

```
rangeLow(op)
```

Arguments

op	an opAttribute object that is either numeric or integer that defines the logical operation used to determine eligible range low

Value

An attribute that can be used in a cohort definition

readConceptSet *Read a concept set json or csv into R*

Description

Concept sets can be serialized to json or csv file formats. ‘readConceptSet‘ reads the files into R as Capr concepts sets.

Usage

```
readConceptSet(path, name, id = NULL)
```

Arguments

path	Name of concept set file to read in csv or json format. (e.g. "concepts.json")
name	the name of the concept set
id	the id for the concept set (keep?)

Examples

```
library(Capr)
path <- tempfile("concepts", fileext = ".json")
concepts <- cs(1, 2, descendants(4, 5), exclude(descendants(6, 7)), name = "test")
writeConceptSet(concepts, path = path)

concepts <- readConceptSet(path)
```

startDate *Function that creates a start date attribute*

Description

Function that creates a start date attribute

Usage

```
startDate(op, type = "occurrence")
```

Arguments

op	an opAttribute object must be a date that defines the logical operation used to determine eligible start dates
type	specify the type of date to use either occurrence or era. default as occurrence

Value

An attribute that can be used in a cohort definition

toCirce *Function to coerce cohort to circe*

Description

Function to coerce cohort to circe

Usage

```
toCirce(cd)
```

Arguments

cd	the Capr cohort class

Value

an s3 list representing the circe object to coerce to json

unit *Add unit attribute to a query*

Description

Add unit attribute to a query

Usage

```
unit(x)
```

Arguments

x	A single character idetifier for a unit or a concept set that identifies units

Value

An attribute that can be used in a query function

Examples

```
# create a unit attribute
unit(8713L)
unit("%")
```

valueAsNumber *Function to create valueAsNumber attribute*

Description

This function is used only for measurement query. valueAsNumber is a column in the measurement table of the cdm. This attribute allows a subquery to find measurements that satisfy certain values determined by the op logic.

Usage

```
valueAsNumber(op)
```

Arguments

op	an opAttribute object that is either numeric or integer that defines the logical operation used to determine eligible patient age

Value

An attribute that can be used in a cohort definition

visit *Query the visit occurrence domain*

Description

Query the visit occurrence domain

Usage

```
visit(conceptSet, ...)
```

Arguments

conceptSet	A condition concept set
...	optional attributes

Value

A Capr Query

visitType — *Add a visit type attribute to determine the provenance of the record*

Description

Add a visit type attribute to determine the provenance of the record

Usage

```
visitType(ids, connection, vocabularyDatabaseSchema)
```

Arguments

ids	the concept ids for the attribute
connection	a connection to an OMOP dbms to get vocab info about the concept
vocabularyDatabaseSchema	the database schema for the vocabularies

Value

An attribute that can be used in a query function

withAll — *Function to construct a group where all criteria and groups must be satisfied*

Description

Function to construct a group where all criteria and groups must be satisfied

Usage

```
withAll(...)
```

Arguments

...	a set of criteria or groups

Value

an S4 group class specifying a bundle of criteria that all must be satisfied in context of the cohort definition

`withAny` *Function to construct a group where any criteria and groups may be satisfied*

Description

Function to construct a group where any criteria and groups may be satisfied

Usage

```
withAny(...)
```

Arguments

`...`	a set of criteria or groups

Value

an S4 group class specifying a bundle of criteria that any may be satisfied in context of the cohort definition

`withAtLeast` *Function to construct a group where at least some of the criteria or groups must be satisfied*

Description

Function to construct a group where at least some of the criteria or groups must be satisfied

Usage

```
withAtLeast(x, ...)
```

Arguments

`x`	an integer specifying the number of criteria or groups that must be satisfied
`...`	a set of criteria or groups

Value

an S4 group class specifying a bundle of criteria that at least x be satisfied in context of the cohort definition

withAtMost — *Function to construct a group where at most some of the criteria or groups must be satisfied*

Description

Function to construct a group where at most some of the criteria or groups must be satisfied

Usage

```
withAtMost(x, ...)
```

Arguments

x	an integer specifying the number of criteria or groups that must be satisfied
...	a set of criteria or groups

Value

an S4 group class specifying a bundle of criteria that at most x be satisfied in context of the cohort definition

writeCohort — *Write Cohort json file*

Description

Write Cohort json file

Usage

```
writeCohort(x, path)
```

Arguments

x	A Capr cohort
path	The name of the file to create

Value

Invisibly returns the path to the json file that was written

Examples

```
## Not run:
cs1 <- cs(descendants(exclude(436665),440383,442306,4175329), name = "concepts")
# optional step to fill in concept set details. Requires database connection.
con <- {A CDM datbase connection}
cs1 <- getConceptSetDetails(cs1, con)

x <- cohort(conditionOccurrence(cs1))
writeCohort(x, "cohortDefinition.json")

## End(Not run)
```

`writeConceptSet` *Save a concept set as a json file*

Description

The resulting concept Set JSON file can be imported into Atlas.

Usage

```
writeConceptSet(x, path, format = "auto", ...)
```

Arguments

`x`	A Capr concept set created by ‘cs()‘
`path`	Name of file to write to. (e.g. "concepts.json")
`format`	the file extension to write
`...`	additional arguments

Examples

```
anemia <- cs(descendants(439777,4013073,4013074), name = "anemia")
writeConceptSet(anemia, file.path(tempdir(), 'anemia.json'))
writeConceptSet(anemia, file.path(tempdir(), 'anemia.csv'))
```

Index