Using Capr

Contents

Capr provides R users with a language for building computable cohort definitions; i.e. definitions that can be translated to SQL code and executed on a observational health database in the OMOP Common Data Model format.

While some familiarity with the OMOP Common Data Model standard and the Observational Health Data Science and Informatics (OHDSI) open source ecosystem is recommended, this tutorial assumes very little prior knowledge.

We define a "cohort" as a set of persons who meet a set of inclusion criteria for a duration of time. A cohort can be thought of a set of person-time, the time during which persons in a database met the cohort definition.

A cohort is built by identifying a set of potential index dates and then applying inclusion criteria to those potential index dates. Finally persons exit a cohort either at an explicitly specified time (e.g. X days after index) or when they are no longer under observation.

In the OMOP CDM data elements are standardized so every OMOP CDM database uses the same codes to represent identical clinical data elements. Codes can be looked up in a publicly available vocabulary search tool called Athena.

Let's jump into some simple cohort definitions with Capr. We can use an example database called `Eunomia` to create reproducible code examples.

The condition concept ID for Gastrointestinal hemorrhage is 192671. We can create a concept set with the `cs` function which works similar to the `c` function for creating vectors in R.

```
library(Capr)

GIBleed <- cs(descendants(192671))

GIBleed
```

The GIBleed concept set will include all descendants of the Gastrointestinal hemorrhage concept 192671.

Creating a simple cohort with the index date at the first occurrence of a GI bleed condition occurrence is done in a single line of R code. This cohort takes advantage of many defaults that match the defaults in Atlas.

```
giBleedCohort <- cohort(condition(GIBleed))

giBleedCohort
```

Cohort generation is the process of converting a cohort definition to a table with four columns: cohort_definition_id, subject_id, cohort_start_date, cohort_end_date. This table contains the person IDs and dates that people are in the cohort.

```


connectionDetails <- Eunomia::getEunomiaConnectionDetails()
```

```r
sql <- CirceR::buildCohortQuery(CirceR::cohortExpressionFromJson(as.json(giBleedCohort)),
                                CirceR::createGenerateOptions(
                                  generateStats = FALSE
                                ))

cohortsToCreate <- tibble::tibble(
  cohortId = 1,
  cohortName = "GI Bleed",
  sql = sql
)


cohortTableNames <- CohortGenerator::getCohortTableNames(cohortTable = "my_cohort_table")
CohortGenerator::createCohortTables(connectionDetails = connectionDetails,
                                                         cohortDatabaseSchema = "main",
                                                         cohortTableNames = cohortTableNames)
# Generate the cohorts
cohortsGenerated <- CohortGenerator::generateCohortSet(connectionDetails = connectionDetails,
                                                       cdmDatabaseSchema = "main",
                                                       cohortDatabaseSchema = "main",
                                                       cohortTableNames = cohortTableNames,
                                                       cohortDefinitionSet = cohortsToCreate)

# Get the cohort counts
cohortCounts <- CohortGenerator::getCohortCounts(connectionDetails = connectionDetails,
                                                 cohortDatabaseSchema = "main",
                                                 cohortTable = cohortTableNames$cohortTable)
print(cohortCounts)
sql <- SqlRender::render(sql)
sql <- SqlRender::translate(sql, "sqlite")

con <- DatabaseConnector::connect(connectionDetails)

DatabaseConnector::executeSql(con, sql)
```