Using CohortIncidence

Christopher Knoll

2021-08-27

Contents

1 Introduction

This vignette describes how to use the `CohortIncidence` package to perform a single incidence rate analysis for a given target and outcome cohort, with a few settings for Time At Risk and Clean Window.

2 Installation instructions

Before installing the `CohortIncidence` package make sure you have Java available. Java can be downloaded from www.java.com. For Windows users, RTools is also necessary. RTools can be downloaded from CRAN.

The `CohortIncidence` package is currently maintained in a Github repository.

```r
install.packages("remotes")
remotes::install_github("ohdsi/CohortIncidence")
```

Once installed, you can type `library(CohortIncidence)` to load the package.

3 Database Preparation

The results of the anlaysis SQL will assume a final table: `@results_database_schema.incidence_summary`. The DDL for this table can be fetched from the package via the following:

```
# Fetch DDL from package
ddl <- CohortIncidence::getResultsDdl();
cat(ddl)
CREATE TABLE @schemaName.incidence_summary
(
    ref_id int,
    database_name varchar(255),
    target_cohort_definition_id bigint,
    target_name varchar(255),
    time_at_risk_id bigint,
    time_at_risk_start_offset bigint,
    time_at_risk_start_index bigint,
    time_at_risk_end_offset bigint,
    time_at_risk_end_index bigint,
    subgroup_id bigint,
    subgroup_name varchar(255),
    outcome_id bigint,
    outcome_cohort_definition_id bigint,
    outcome_name varchar(255),
    clean_window bigint,
    persons_pre_exclude bigint,
    persons_at_risk bigint,
    person_days_pre_exclude bigint,
    person_days bigint,
    person_outcomes_pre_exclude bigint,
    person_outcomes bigint,
    outcomes_pre_exclude bigint,
    outcomes bigint,
    incidence_proportion_p100p float,
    incidence_rate_p100py float
 );
```

Using `SqlRender` and `DatabaseConnector`, you can execute the above on your target database platform in order to deploy the table. Remember to replace `@schemaName` with the appropriate schema. You can also 'hack' the `@schemaName` paramater to apply a prefix by specifying the SqlRender paramater of `schemaName.incidence_summary` to a target table ie: `mySchema.prefix_incidence_summary`. Using the same paramater name/value in `buildQuery()` will allow you to provide a prefix to the result table name instead of having to declare a separate schema.

```
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "postgresql",server={Sys.getenv(

# to specify the target schema (the typical use case):
ddl <- SqlRender::render(CohortIncidence::getResultsDdl(), schemaName = "mySchema");

# a work-around to provide a prefix to the result table, in case creating new schema is restricted
ddlPrefix <- SqlRender::render(CohortIncidence::getResultsDdl(), "schemaName.incidence_summary" = "mySch

con <- DatabaseConnector::connect(connectionDetails);
```

```
DatabaseConnector::executeSql(ddl);
DatabaseConnector::disconnect(con);
```

4 A simple example

This example will create a CohortIncidence design containing a single target, outcome, and time at risk.

4.1 Build the design

The following script builds a single T, O and Time at Risk, and assembles those element into a design. Finally, the resulting JSON is printed.

```
t1 <- CohortIncidence::createCohortRef(id=1, name="Target cohort 1");

o1 <- CohortIncidence::createOutcomeDef(id=1,name="Outcome 1, 30d Clean",
                                               cohortId =2,
                                               cleanWindow =30);

tar1 <- CohortIncidence::createTimeAtRiskDef(id=1,
                                             startDateField="StartDate",
                                             endDateField="StartDate",
                                             endOffset=30);

# Note: c() is used when dealing with an array of numbers,
# later we use list() when dealing with an array of objects
analysis1 <- CohortIncidence::createIncidenceAnalysis(targets = c(t1$id),
                                                      outcomes = c(o1$id),
                                                      tars = c(tar1$id));

subgroup1 <- CohortIncidence::createCohortSubgroup(id=1, name="Subgroup 1", cohortRef = createCohortRef


# Create Design (note use of list() here):
irDesign <- CohortIncidence::createIncidenceDesign(targetDefs = list(t1),
                                                   outcomeDefs = list(o1),
                                                   tars=list(tar1),
                                                   analysisList = list(analysis1),
                                                   subgroups = list(subgroup1));
# Render the design as JSON
jsonlite::toJSON(irDesign,pretty = T)
#> {
#>   "cohortDefs": [],
#>   "targetDefs": [
#>     {
#>       "id": 1,
#>       "name": "Target cohort 1"
#>     }
#>   ],
#>   "outcomeDefs": [
#>     {
#>       "id": 1,
#>         "name": "Outcome 1, 30d Clean",
#>         "cohortId": 2,
#>         "cleanWindow": 30
#>       }
#>     ],
#>     "timeAtRiskDefs": [
#>       {
#>         "id": 1,
#>         "start": {
#>           "dateField": "StartDate",
#>           "offset": 0
#>         },
#>         "end": {
#>           "dateField": "StartDate",
#>           "offset": 30
#>         }
#>       }
#>     ],
#>     "analysisList": [
#>       {
#>         "targets": [1],
#>         "outcomes": [1],
#>         "tars": [1]
#>       }
#>     ],
#>     "conceptSets": [],
#>     "subgroups": [
#>       {
#>         "CohortSubgroup": {
#>           "id": 1,
#>           "name": "Subgroup 1",
#>           "cohort": {
#>             "id": 300
#>           }
#>         }
#>       }
#>     ]
#> }
```

4.2 Build analysis SQL from design

From the previous design, the `CohortIncidence::buildQuery()` method is used to generate the analysis SQL:

```
buildOptions <- CohortIncidence::buildOptions(cohortTable = "demoCohortSchema.cohort",
                                              cdmSchema = "mycdm",
                                              resultsSchema = "myresults",
                                              refId = 1)

analysisSql <- CohortIncidence::buildQuery(incidenceDesign =  as.character(jsonlite::toJSON(irDesign)),
                                           buildOptions = buildOptions);
cat(analysisSql)
```

```
#> select target_cohort_definition_id, target_name
#> into #target_ref
#> from (
#> select cast(1 as int) as target_cohort_definition_id,
#>  cast ('Target cohort 1' as varchar(250)) as target_name
#> ) O
#> ;
#>
#> select time_at_risk_id, time_at_risk_start_index, time_at_risk_start_offset, time_at_risk_end_index,
#> into #tar_ref
#> FROM (
#> select cast(1 as int) as time_at_risk_id,
#>  cast (1 as int) as time_at_risk_start_index, cast (0 as int) as time_at_risk_start_offset,
#>  cast (1 as int) as time_at_risk_end_index, cast (30 as int) as time_at_risk_end_offset
#> ) T
#> ;
#>
#> select outcome_id, outcome_cohort_definition_id, outcome_name, clean_window, excluded_cohort_definit
#> into #outcome_ref
#> from (
#> select cast(1 as int) as outcome_id, cast(2 as int) as outcome_cohort_definition_id,
#>  cast ('Outcome 1, 30d Clean' as varchar(255)) as outcome_name,
#>  cast (30 as int) as clean_window,
#>   cast (0 as int) as excluded_cohort_definition_id
#> ) O
#> ;
#>
#> -- Will figure out subgroup mechanics at a later time
#> select subgroup_id, subgroup_name
#> INTO #subgroup_ref
#> FROM (
#> select cast(0 as int) as subgroup_id,
#>  cast ('All' as varchar(250)) as subgroup_name
#> UNION ALL
#> select cast(1 as int) as subgroup_id,
#>  cast ('Subgroup 1' as varchar(250)) as subgroup_name
#> ) S
#> ;
#>
#> --
#> -- Begin analysis 0
#> --
#>
#> /****************************************
#> code to implement calculation using the inputs above, no need to modify beyond this point
#>
#> 1) create T + TAR periods
#> 2) determine which TTAR periods require era-fying, and which don't
#> 3) create table to store era-fied at-risk periods
#>   put all periods that don't require erafying
#>   era-fy those records that require it, then put them in table
#> 4) create the exc_o periods,  per TTAR
#> 5) create table to sore era-fied exc_at_risk periods
#>   put all periods that don't require erafying
#>   era-fy those records that require it, then put them in table
#> 6) overall, T/O/TAR,  compute TAR = sum(at-risk_era) - sum(exc_at_risk_era),  num_events = sum(events
#> 7) join to S and C, compute T/S/C/O/TAR   person-time, num_events
#>
#> *************************************/
#>
#> --three ways for entry into excluded
#> --1:  duration of outcome periods  (ex:  immortal time due to clean period)
#> --2:  other periods excluded  (ex: persons post-appendectomy for appendicitis)
#> --3:  if you wanted to exclude persons with prior events or set to '1st event only' (set exclusion f
#>
#> --HINT DISTRIBUTE_ON_KEY(subject_id)
#> select cohort_definition_id, time_at_risk_id, subject_id, start_date, end_date
#> into #TTAR
#> FROM (
#>  select tc1.cohort_definition_id,
#>      tar1.time_at_risk_id,
#>      subject_id,
#>      case
#>          when tar1.time_at_risk_start_index = 1 then
#>              case when dateadd(dd,tar1.time_at_risk_start_offset,tc1.cohort_start_date) < op1.observ
#>                  when dateadd(dd,tar1.time_at_risk_start_offset,tc1.cohort_start_date) >= op1.observ
#>              end
#>          when tar1.time_at_risk_start_index = 0 then
#>              case when dateadd(dd,tar1.time_at_risk_start_offset,tc1.cohort_end_date) < op1.observat
#>                  when dateadd(dd,tar1.time_at_risk_start_offset,tc1.cohort_end_date) >= op1.observat
#>              end
#>          else null --shouldnt get here if tar set properly
#>      end as start_date,
#>      case
#>          when tar1.time_at_risk_end_index = 1 then
#>              case when dateadd(dd,tar1.time_at_risk_end_offset,tc1.cohort_start_date) < op1.observat
#>                  when dateadd(dd,tar1.time_at_risk_end_offset,tc1.cohort_start_date) >= op1.observat
#>              end
#>          when tar1.time_at_risk_end_index = 0 then
#>              case when dateadd(dd,tar1.time_at_risk_end_offset,tc1.cohort_end_date) < op1.observatio
#>                  when dateadd(dd,tar1.time_at_risk_end_offset,tc1.cohort_end_date) >= op1.observatio
#>              end
#>          else null --shouldnt get here if tar set properly
#>      end as end_date
#>  from (select time_at_risk_id, time_at_risk_start_index, time_at_risk_start_offset, time_at_risk_end
#>  (select cohort_definition_id, subject_id, cohort_start_date, cohort_end_date from demoCohortSchema.
#>  inner join mycdm.observation_period op1 on tc1.subject_id = op1.person_id
#>      and tc1.cohort_start_date >= op1.observation_period_start_date
#>      and tc1.cohort_start_date <= op1.observation_period_end_date
#> ) TAR
#> WHERE TAR.start_date <= TAR.end_date
#> ;
#>
#> --find the records that need to be era-fied
#>
#> --HINT DISTRIBUTE_ON_KEY(subject_id)
```

```
#>
#>
#> --era-building script for the 'TTAR_to_erafy' records
#> --insert records from era-building script into #TTAR_erafied
#> --HINT DISTRIBUTE_ON_KEY(subject_id)
#> select t1.cohort_definition_id, t1.time_at_risk_id, t1.subject_id, t1.start_date, t1.end_date
#> INTO #TTAR_to_erafy
#> from #TTAR t1
#> inner join #TTAR t2 on t1.cohort_definition_id = t2.cohort_definition_id
#>  and t1.time_at_risk_id = t2.time_at_risk_id
#>  and t1.subject_id = t2.subject_id
#>  and t1.start_date <= t2.end_date
#>  and t1.end_date >= t2.start_date
#>  and t1.start_date <> t2.start_date
#> ;
#>
#> --HINT DISTRIBUTE_ON_KEY(subject_id)
#> with cteEndDates (cohort_definition_id, time_at_risk_id, subject_id, end_date) AS
#> (
#>  SELECT
#>        cohort_definition_id,
#>          time_at_risk_id,
#>          subject_id,
#>        event_date as end_date
#>  FROM
#>  (
#>      SELECT cohort_definition_id,
#>          time_at_risk_id,
#>          subject_id,
#>          event_date,
#>          SUM(event_type) OVER (PARTITION BY cohort_definition_id, time_at_risk_id, subject_id ORDER
#>      FROM
#>      (
#>          SELECT
#>              cohort_definition_id,
#>              time_at_risk_id,
#>              subject_id,
#>              start_date AS event_date,
#>            -1 AS event_type
#>          FROM #TTAR_to_erafy
#>
#>          UNION ALL
#>
#>          SELECT
#>              cohort_definition_id,
#>              time_at_risk_id,
#>              subject_id,
#>              end_date AS event_date,
#>            1 AS event_type
#>          FROM #TTAR_to_erafy
#>      ) RAWDATA
#>  ) e
#>  WHERE interval_status = 0
#> ),
#> cteEnds (cohort_definition_id, time_at_risk_id, subject_id, start_date, end_date) AS
#> (
#>  SELECT c.cohort_definition_id,
#>      c.time_at_risk_id,
#>      c.subject_id,
#>      c.start_date,
#>      MIN(e.end_date) AS end_date
#>  FROM #TTAR_to_erafy c
#>  INNER JOIN cteEndDates e ON c.subject_id = e.subject_id
#>      AND c.cohort_definition_id = e.cohort_definition_id
#>      AND c.time_at_risk_id = e.time_at_risk_id
#>      AND e.end_date >= c.start_date
#>  GROUP BY  c.cohort_definition_id,
#>      c.time_at_risk_id,
#>      c.subject_id,
#>      c.start_date
#> )
#> select cohort_definition_id, time_at_risk_id, subject_id, min(start_date) as start_date, end_date
#> into #TTAR_era_overlaps
#> from cteEnds
#> group by cohort_definition_id, time_at_risk_id, subject_id, end_date
#> ;
#>
#>
#> --HINT DISTRIBUTE_ON_KEY(subject_id)
#> select cohort_definition_id, cast(0 as int) as subgroup_id, time_at_risk_id, subject_id, start_date,
#> into #TTAR_erafied
#> FROM (
#>  select cohort_definition_id, time_at_risk_id, subject_id, start_date, end_date
#>  from #TTAR_era_overlaps
#>
#>  UNION ALL
#>
#>  --records that were already erafied and just need to be brought over directly
#>  select distinct t1.cohort_definition_id, t1.time_at_risk_id, t1.subject_id, t1.start_date, t1.end_d
#>  from #TTAR t1
#>  left join #TTAR t2 on t1.cohort_definition_id = t2.cohort_definition_id
#>      and t1.time_at_risk_id = t2.time_at_risk_id
#>      and t1.subject_id = t2.subject_id
#>      and t1.start_date <= t2.end_date
#>      and t1.end_date >= t2.start_date
#>      and t1.start_date <> t2.start_date
#>  where t2.subject_id IS NULL
#> ) T
#>
#> ;
#>
#> INSERT INTO #TTAR_erafied (cohort_definition_id, subgroup_id, time_at_risk_id, subject_id, start_dat
#> select t1.cohort_definition_id, cast(1 as int) as subgroup_id, t1.time_at_risk_id, t1.subject_id, t1
#> FROM #TTAR_erafied t1
#> JOIN demoCohortSchema.cohort s1 on t1.subject_id = s1.subject_id
#>   and t1.start_date  >= s1.cohort_start_date
#>   and t1.start_date <= s1.cohort_end_date
#> WHERE s1.cohort_definition_id = 300
#> ;
#>
#>
#> -- find excluded time from outcome cohorts
#> -- note, clean window added to event end date
#> --HINT DISTRIBUTE_ON_KEY(subject_id)
#> select or1.outcome_id, oc1.subject_id, dateadd(dd,1,oc1.cohort_end_date) as cohort_start_date, datea
#> into #excluded_tar_cohort
#> from demoCohortSchema.cohort oc1
#> inner join (
#>  select outcome_id, outcome_cohort_definition_id, clean_window
#>  from #outcome_ref
#>  where outcome_id in (1)
#> ) or1 on oc1.cohort_definition_id = or1.outcome_cohort_definition_id
#> where dateadd(dd,or1.clean_window, oc1.cohort_end_date) >= dateadd(dd,1,oc1.cohort_end_date)
#>
#> union all
#>
#> SELECT or1.outcome_id, c1.subject_id, c1.cohort_start_date, c1.cohort_end_date
#> FROM demoCohortSchema.cohort c1
#> inner join (
#>  select outcome_id, excluded_cohort_definition_id
#>  from #outcome_ref
#>  where outcome_id in (1)
#> ) or1 on c1.cohort_definition_id = or1.excluded_cohort_definition_id
#> ;
#>
#> --HINT DISTRIBUTE_ON_KEY(subject_id)
#> select te1.cohort_definition_id as target_cohort_definition_id,
#>  te1.time_at_risk_id,
#>  ec1.outcome_id,
#>  ec1.subject_id,
#>  case when ec1.cohort_start_date > te1.start_date then ec1.cohort_start_date else te1.start_date end
#>  case when ec1.cohort_end_date < te1.end_date then ec1.cohort_end_date else te1.end_date end as end_
#> into #exc_TTAR_o
#> from #TTAR_erafied te1
#> inner join #excluded_tar_cohort ec1 on te1.subject_id = ec1.subject_id
#>  and ec1.cohort_start_date <= te1.end_date
#>  and ec1.cohort_end_date >= te1.start_date
#> ;
#>
#> --find the records that need to be era-fied
#>
#> --HINT DISTRIBUTE_ON_KEY(subject_id)
#> select t1.target_cohort_definition_id, t1.time_at_risk_id, t1.outcome_id, t1.subject_id, t1.start_da
#> into #exc_TTAR_o_to_erafy
#> from #exc_TTAR_o t1
#> inner join #exc_TTAR_o t2 on t1.target_cohort_definition_id = t2.target_cohort_definition_id
#>   and t1.time_at_risk_id = t2.time_at_risk_id
#>   and t1.outcome_id = t2.outcome_id
#>   and t1.subject_id = t2.subject_id
#>   and t1.start_date < t2.end_date
#>   and t1.end_date > t2.start_date
#>   and (t1.start_date <> t2.start_date or t1.end_date <> t2.end_date)
#> ;
#>
#> --era-building script for the 'exc_TTAR_o_to_erafy ' records
#> --insert records from era-building script into #TTAR_erafied
#>
#> --HINT DISTRIBUTE_ON_KEY(subject_id)
#> with cteEndDates (target_cohort_definition_id, time_at_risk_id, outcome_id, subject_id, end_date) AS
#> (
#>  SELECT
#>        target_cohort_definition_id,
#>          time_at_risk_id,
#>          outcome_id,
#>          subject_id,
#>        event_date as end_date
#>  FROM
#>  (
#>      SELECT
#>          target_cohort_definition_id,
#>              time_at_risk_id,
#>              outcome_id,
#>              subject_id,
#>              event_date,
#>              SUM(event_type) OVER (PARTITION BY target_cohort_definition_id, time_at_risk_id, outcom
#>      FROM
#>      (
#>          SELECT
#>              target_cohort_definition_id,
#>              time_at_risk_id,
#>              outcome_id,
#>              subject_id,
#>              start_date AS event_date,
#>            -1 AS event_type
#>          FROM #exc_TTAR_o_to_erafy
#>
#>          UNION ALL
#>
#>          SELECT
#>              target_cohort_definition_id,
#>              time_at_risk_id,
#>              outcome_id,
#>              subject_id,
#>              end_date AS event_date,
#>            1 AS event_type
#>          FROM #exc_TTAR_o_to_erafy
#>      ) RAWDATA
#>  ) e
#>  WHERE interval_status = 0
#> ),
#> cteEnds (target_cohort_definition_id, time_at_risk_id, outcome_id, subject_id, start_date, end_date)
#> (
#>  SELECT c.target_cohort_definition_id,
#>   c.time_at_risk_id,
#>   c.outcome_id,
#>       c.subject_id,
#>      c.start_date,
#>      MIN(e.end_date) AS end_date
#>  FROM #exc_TTAR_o_to_erafy c
#>  INNER JOIN cteEndDates e
#>   ON c.subject_id = e.subject_id
#>   AND c.target_cohort_definition_id = e.target_cohort_definition_id
#>   AND c.time_at_risk_id = e.time_at_risk_id
#>   AND c.outcome_id = e.outcome_id
#>   AND e.end_date >= c.start_date
#>  GROUP BY  c.target_cohort_definition_id,
#>   c.time_at_risk_id,
#>   c.outcome_id,
#>       c.subject_id,
#>      c.start_date
#> )
#> select target_cohort_definition_id, time_at_risk_id, outcome_id, subject_id, min(start_date) as star
#> into #ex_TTAR_o_overlaps
#> from cteEnds
#> group by target_cohort_definition_id, time_at_risk_id, outcome_id, subject_id, end_date
#> ;
#>
#> --HINT DISTRIBUTE_ON_KEY(subject_id)
#> select target_cohort_definition_id, time_at_risk_id, outcome_id, subject_id, start_date, end_date
#> into #exc_TTAR_o_erafied
#> from #ex_TTAR_o_overlaps
#>
#> UNION ALL
#>
#> --records that were already erafied and just need to be brought over directly
#> select distinct t1.target_cohort_definition_id, t1.time_at_risk_id, t1.outcome_id, t1.subject_id, t1
#> from #exc_TTAR_o t1
#> left join #exc_TTAR_o t2 on t1.target_cohort_definition_id = t2.target_cohort_definition_id
#>   and t1.time_at_risk_id = t2.time_at_risk_id
#>   and t1.outcome_id = t2.outcome_id
#>   and t1.subject_id = t2.subject_id
#>   and t1.start_date < t2.end_date
#>   and t1.end_date > t2.start_date
#>   and (t1.start_date <> t2.start_date or t1.end_date <> t2.end_date)
#> where t2.subject_id IS NULL
#> ;
#>
#>
#> --calculate time_at_risk
#>
#> create table #at_risk_smry_pre_xcl
#> (
#>   target_cohort_definition_id bigint,
#>   time_at_risk_id int,
#>   subgroup_id bigint,
#>   num_persons bigint,
#>   person_days bigint
#> )
#> ;
#>
#>
#> INSERT INTO #at_risk_smry_pre_xcl (target_cohort_definition_id, time_at_risk_id,subgroup_id, num_pers
#> select t1.cohort_definition_id as target_cohort_definition_id,
#>   t1.time_at_risk_id,
#>  t1.subgroup_id,
#>   count_big(distinct t1.subject_id) as num_persons,
#>   sum((datediff(dd,t1.start_date, t1.end_date)+1)) as person_days
#> from #TTAR_erafied t1
#> group by t1.cohort_definition_id, t1.subgroup_id, t1.time_at_risk_id
#> ;
#>
#> --calculate events during pre_exclude at risk
#>
#> create table #outcome_smry_pre_xcl
#> (
#>   target_cohort_definition_id bigint,
#>   time_at_risk_id int,
#>   subgroup_id bigint,
#>   outcome_id bigint,
#>   num_person_outcomes bigint,
#>   num_outcomes bigint
#> )
#> ;
#>
#> insert into #outcome_smry_pre_xcl (target_cohort_definition_id, time_at_risk_id,subgroup_id, outcome_
#> select t1.cohort_definition_id as target_cohort_definition_id,
#>   t1.time_at_risk_id,
#>   t1.subgroup_id,
#>   o1.outcome_id,
#>   count_big(distinct o1.subject_id) as num_person_outcomes,
#>   count_big(o1.subject_id) as num_outcomes
#> from #TTAR_erafied t1
#> inner join (
#>  select oref.outcome_id, oc.subject_id, oc.cohort_start_date, oc.cohort_end_date
#>  from demoCohortSchema.cohort oc
#>  JOIN #outcome_ref oref on oc.cohort_definition_id = oref.outcome_cohort_definition_id
#>  where oref.outcome_id in (1)
#> ) o1 on t1.subject_id = o1.subject_id
#>  and t1.start_date <= o1.cohort_start_date
#>  and t1.end_date >= o1.cohort_start_date
#> group by t1.cohort_definition_id, t1.subgroup_id, t1.time_at_risk_id, o1.outcome_id
#> ;
#>
#> --4 statistics to calculate to exclude:
#>   --1. person_days to exclude
#>   --2. num persons w no tar after exclusion
#>   --3. num outcomes to exclude
#>   --4. num persons w no outcome after exclusion
```

```
#>
#> --1. person_days to exclude
#>
#> create table #excluded_person_days
#> (
#>   target_cohort_definition_id bigint,
#>   time_at_risk_id int,
#>   subgroup_id bigint,
#>   outcome_id bigint,
#>   person_days bigint
#> )
#> ;
#>
#>
#> INSERT INTO #excluded_person_days (target_cohort_definition_id, time_at_risk_id,subgroup_id, outcome_
#> select et1.target_cohort_definition_id,
#>   et1.time_at_risk_id,
#>   t1.subgroup_id,
#>   et1.outcome_id,
#>   sum(datediff(dd,et1.start_date, et1.end_date) + 1) as person_days
#> from #TTAR_erafied t1
#> inner join #exc_TTAR_o_erafied et1 on t1.cohort_definition_id = et1.target_cohort_definition_id
#>   and t1.time_at_risk_id = et1.time_at_risk_id
#>   and t1.subject_id = et1.subject_id
#>   and t1.start_date <= et1.start_date
#>   and t1.end_date >= et1.end_date
#> group by et1.target_cohort_definition_id, et1.time_at_risk_id, t1.subgroup_id, et1.outcome_id
#> ;
#>
#>
#> --2. num persons w no tar after exclusion
#> --find persons with >=1d at-risk  (T - exc) > 0
#>
#> create table #excluded_persons
#> (
#>   target_cohort_definition_id bigint,
#>   time_at_risk_id int,
#>   subgroup_id bigint,
#>   outcome_id  bigint,
#>   num_persons_w_no_tar bigint
#> );
#>
#> insert into #excluded_persons (target_cohort_definition_id, time_at_risk_id, subgroup_id, outcome_id
#> select t1.target_cohort_definition_id,
#>   t1.time_at_risk_id,
#>   t1.subgroup_id,
#>   et1.outcome_id,
#>   count_big(distinct t1.subject_id) as num_persons_w_no_tar
#> from
#> (
#>   select t0.cohort_definition_id as target_cohort_definition_id,
#>     t0.time_at_risk_id,
#>      t0.subgroup_id,
#>     t0.subject_id,
#>     sum(datediff(dd,t0.start_date,t0.end_date)) as person_days
#>   from #TTAR_erafied t0
#>   inner join (select distinct target_cohort_definition_id, subject_id from #exc_TTAR_o_erafied) e0 o
#>      and t0.cohort_definition_id = e0.target_cohort_definition_id
#>   group by t0.cohort_definition_id,
#>     t0.time_at_risk_id,
#>      t0.subgroup_id,
#>     t0.subject_id
#> ) t1
#> inner join
#> (
#>   select target_cohort_definition_id,
#>     time_at_risk_id,
#>     outcome_id,
#>     subject_id,
#>     sum(datediff(dd,start_date,end_date)) as person_days
#>   from #exc_TTAR_o_erafied
#>   group by target_cohort_definition_id,
#>     time_at_risk_id,
#>     outcome_id,
#>     subject_id
#> ) et1
#>   on t1.subject_id = et1.subject_id
#>   and t1.target_cohort_definition_id = et1.target_cohort_definition_id
#>   and t1.time_at_risk_id = et1.time_at_risk_id
#>   and t1.person_days = et1.person_days
#> group by t1.target_cohort_definition_id,
#>   t1.time_at_risk_id,
#>  t1.subgroup_id,
#>   et1.outcome_id
#> ;
#>
#> --3. num outcomes to exclude
#> --calculate events during pre_exclude at risk
#>
#> create table #excluded_outcomes
#> (
#>   target_cohort_definition_id bigint,
#>   time_at_risk_id int,
#>   subgroup_id bigint,
#>   outcome_id bigint,
#>   num_outcomes bigint
#> )
#> ;
#>
#>
#> insert into #excluded_outcomes (target_cohort_definition_id, time_at_risk_id,subgroup_id,outcome_id,
#> select et1.target_cohort_definition_id,
#>   et1.time_at_risk_id,
#>   t1.subgroup_id,
#>   et1.outcome_id,
#>   count_big(o1.subject_id) as num_outcomes
#> from #TTAR_erafied t1
#> inner join #exc_TTAR_o_erafied et1 on t1.cohort_definition_id = et1.target_cohort_definition_id
#>   and t1.time_at_risk_id = et1.time_at_risk_id
#>   and t1.subject_id = et1.subject_id
#>   and t1.start_date <= et1.start_date
#>   and t1.end_date >= et1.end_date
#> inner join (
#>  select oref.outcome_id, oc.subject_id, oc.cohort_start_date, oc.cohort_end_date
#>  from demoCohortSchema.cohort oc
#>  JOIN #outcome_ref oref on oc.cohort_definition_id = oref.outcome_cohort_definition_id
#>  where oref.outcome_id in (1)
#> ) o1 on et1.subject_id = o1.subject_id
#>  and et1.outcome_id = o1.outcome_id
#>  and et1.start_date <= o1.cohort_start_date
#>  and et1.end_date >= o1.cohort_start_date
#> group by et1.target_cohort_definition_id, et1.time_at_risk_id, t1.subgroup_id, et1.outcome_id
#> ;
#>
#> --4. num persons w no outcome after exclusion
#>
#> create table #excl_persons_w_o
#> (
#>   target_cohort_definition_id bigint,
#>   time_at_risk_id int,
#>   subgroup_id bigint,
#>   outcome_id  bigint,
#>   num_persons_excluded_outcomes bigint
#> );
#>
#> insert into #excl_persons_w_o (target_cohort_definition_id, time_at_risk_id, subgroup_id, outcome_id
#> select t1.target_cohort_definition_id,
#>   t1.time_at_risk_id,
#>   t1.subgroup_id,
#>   t1.outcome_id,
#>   count_big(distinct t1.subject_id) as num_persons_excluded_outcomes
#> from
#> (
#>   select t0.cohort_definition_id as target_cohort_definition_id,
#>     t0.time_at_risk_id,
#>     t0.subject_id,
#>     t0.subgroup_id,
#>     o1.outcome_id,
#>     count(o1.subject_id) as num_outcomes
#>   from #TTAR_erafied t0
#>  inner join (select distinct target_cohort_definition_id, subject_id from #exc_TTAR_o_erafied) e0 on
#>      and t0.cohort_definition_id = e0.target_cohort_definition_id
#>   inner join (
#>      select oref.outcome_id, oc.subject_id, oc.cohort_start_date, oc.cohort_end_date
#>      from demoCohortSchema.cohort oc
#>      JOIN #outcome_ref oref on oc.cohort_definition_id = oref.outcome_cohort_definition_id
#>      where cohort_definition_id in (1)
#>  ) o1 on t0.subject_id = o1.subject_id
#>     and t0.start_date <= o1.cohort_start_date
#>     and t0.end_date >= o1.cohort_start_date
#>   group by  t0.cohort_definition_id,
#>     t0.time_at_risk_id,
#>     t0.subgroup_id,
#>     t0.subject_id,
#>     o1.outcome_id
#> ) t1
#> inner join
#> (
#>   select et1.target_cohort_definition_id,
#>     et1.time_at_risk_id,
#>     et1.subject_id,
#>     et1.outcome_id,
#>     count(o1.subject_id) as num_outcomes
#>   from #TTAR_erafied t1
#>   inner join #exc_TTAR_o_erafied et1 on t1.cohort_definition_id = et1.target_cohort_definition_id
#>     and t1.time_at_risk_id = et1.time_at_risk_id
#>     and t1.subject_id = et1.subject_id
#>     and t1.start_date <= et1.start_date
#>     and t1.end_date >= et1.end_date
#>   inner join (
#>      select oref.outcome_id, oc.subject_id, oc.cohort_start_date, oc.cohort_end_date
#>      from demoCohortSchema.cohort oc
#>      JOIN #outcome_ref oref on oc.cohort_definition_id = oref.outcome_cohort_definition_id
#>      where oref.outcome_id in (1)
#>  ) o1 on et1.subject_id = o1.subject_id
#>     and et1.outcome_id = o1.outcome_id
#>     and et1.start_date <= o1.cohort_start_date
#>     and et1.end_date >= o1.cohort_start_date
#>   group by  et1.target_cohort_definition_id,
#>     et1.time_at_risk_id,
#>     et1.subject_id,
#>     et1.outcome_id
#> ) et1 on t1.subject_id = et1.subject_id
#>   and t1.target_cohort_definition_id = et1.target_cohort_definition_id
#>   and t1.outcome_id = et1.outcome_id
#>   and t1.time_at_risk_id = et1.time_at_risk_id
#>   and t1.num_outcomes = et1.num_outcomes
#> group by t1.target_cohort_definition_id,
#>   t1.time_at_risk_id,
#>   t1.subgroup_id,
#>   t1.outcome_id
#> ;
#>
#> select t1.target_cohort_definition_id,
#>   t1.target_name,
#>  tar1.time_at_risk_id,
#>  tar1.time_at_risk_start_offset,
#>  tar1.time_at_risk_start_index,
#>  tar1.time_at_risk_end_offset,
#>  tar1.time_at_risk_end_index,
#>  s1.subgroup_id,
#>  s1.subgroup_name,
#>  o1.outcome_id,
#>  o1.outcome_cohort_definition_id,
#>  o1.outcome_name,
#>  o1.clean_window
#> into #tscotar_ref
#> from (select * from #target_ref where target_cohort_definition_id in (1))  t1,
#>  (select * from #tar_ref where time_at_risk_id in (1)) tar1,
#>  (select subgroup_id, subgroup_name from #subgroup_ref) s1,
#>  (select * from #outcome_ref where outcome_id in (1)) o1
#> ;
#>
#>
#> select tr1.target_cohort_definition_id,
#>   tr1.target_name,
#>  tr1.time_at_risk_id,
#>  tr1.time_at_risk_start_offset,
#>  tr1.time_at_risk_start_index,
#>  tr1.time_at_risk_end_offset,
#>  tr1.time_at_risk_end_index,
#>  tr1.subgroup_id,
#>  tr1.subgroup_name,
#>  tr1.outcome_id,
#>  tr1.outcome_cohort_definition_id,
#>  tr1.outcome_name,
#>  tr1.clean_window,
#>  coalesce(arspe1.num_persons,0)  as persons_pre_exclude,
#>  coalesce(arspe1.num_persons,0) - coalesce(ep1.num_persons_w_no_tar,0)  as num_persons_at_risk,
#>  coalesce(arspe1.person_days,0) as person_days_pre_exclude,
#>  coalesce(arspe1.person_days,0) - coalesce(epy1.person_days,0) as person_days,
#>  coalesce(ospe1.num_person_outcomes,0) as num_person_outcomes_pre_exclude,
#>  coalesce(ospe1.num_person_outcomes,0) - coalesce(epo1.num_persons_excluded_outcomes,0) as num_perso
#>   coalesce(ospe1.num_outcomes,0) as num_outcomes_pre_exclude,
#>  coalesce(ospe1.num_outcomes,0) - coalesce(eo1.num_outcomes,0) as num_outcomes,
#>  case when coalesce(arspe1.num_persons,0) - coalesce(ep1.num_persons_w_no_tar,0) > 0 then
#>      (100.0 * cast(coalesce(ospe1.num_person_outcomes,0) - coalesce(epo1.num_persons_excluded_outcom
#>      else NULL end as incidence_proportion_p100p,
#>  case when (coalesce(arspe1.person_days,0) - coalesce(epy1.person_days,0)) > 0 then
#>      (100.0 * cast((coalesce(ospe1.num_outcomes,0) - coalesce(eo1.num_outcomes,0)) as float) / ( cas
#>      else NULL end AS incidence_rate_p100py
#> into #incidence_summary
#> from
#> #tscotar_ref tr1
#> left join
#> #at_risk_smry_pre_xcl arspe1
#>  on tr1.target_cohort_definition_id = arspe1.target_cohort_definition_id
#>  and tr1.time_at_risk_id = arspe1.time_at_risk_id
#>  and tr1.subgroup_id = arspe1.subgroup_id
#> left join
#> #outcome_smry_pre_xcl ospe1
#>  on tr1.target_cohort_definition_id = ospe1.target_cohort_definition_id
#>  and tr1.time_at_risk_id = ospe1.time_at_risk_id
#>  and tr1.subgroup_id = ospe1.subgroup_id
#>  and tr1.outcome_id = ospe1.outcome_id
#> left join
#>   #excluded_person_days  epy1
#>     on tr1.target_cohort_definition_id = epy1.target_cohort_definition_id
#>      and tr1.time_at_risk_id = epy1.time_at_risk_id
#>      and tr1.subgroup_id = epy1.subgroup_id
#>      and tr1.outcome_id = epy1.outcome_id
#> left join
#>   #excluded_persons ep1
#>     on tr1.target_cohort_definition_id = ep1.target_cohort_definition_id
#>      and tr1.time_at_risk_id = ep1.time_at_risk_id
#>      and tr1.subgroup_id = ep1.subgroup_id
#>      and tr1.outcome_id = ep1.outcome_id
#> left join
#>   #excluded_outcomes eo1
#>     on tr1.target_cohort_definition_id = eo1.target_cohort_definition_id
#>      and tr1.time_at_risk_id = eo1.time_at_risk_id
#>      and tr1.subgroup_id = eo1.subgroup_id
#>      and tr1.outcome_id = eo1.outcome_id
#> left join
#>   #excl_persons_w_o epo1
#>     on tr1.target_cohort_definition_id = epo1.target_cohort_definition_id
#>      and tr1.time_at_risk_id = epo1.time_at_risk_id
#>      and tr1.subgroup_id = epo1.subgroup_id
#>      and tr1.outcome_id = epo1.outcome_id
#> ;
#>
#> insert into myresults.incidence_summary (ref_id, database_name, target_cohort_definition_id, target_
#>  time_at_risk_id, time_at_risk_start_offset, time_at_risk_start_index, time_at_risk_end_offset, time_
#>  subgroup_id, subgroup_name,
#>  outcome_id, outcome_cohort_definition_id, outcome_name, clean_window,
#>  persons_pre_exclude, persons_at_risk, person_days_pre_exclude, person_days,
#>  person_outcomes_pre_exclude, person_outcomes, outcomes_pre_exclude, outcomes,
#>  incidence_proportion_p100p, incidence_rate_p100py)
#> select CAST(1 as int) as ref_id, '@databaseName' as database_name, is1.target_cohort_definition_id,
#>  is1.time_at_risk_id, is1.time_at_risk_start_offset, is1.time_at_risk_start_index, is1.time_at_risk_
#>  is1.subgroup_id, is1.subgroup_name,
#>  is1.outcome_id, is1.outcome_cohort_definition_id, is1.outcome_name, is1.clean_window,
#>  is1.persons_pre_exclude, is1.num_persons_at_risk, is1.person_days_pre_exclude, is1.person_days,
#>  is1.num_person_outcomes_pre_exclude, is1.num_person_outcomes, is1.num_outcomes_pre_exclude, is1.num_
#>  is1.incidence_proportion_p100p, is1.incidence_rate_p100py
#> from #incidence_summary is1
#> ;
#>
#> -- CLEANUP TEMP TABLES
#> DROP TABLE #excluded_tar_cohort;
#> DROP TABLE #TTAR;
#> DROP TABLE #TTAR_to_erafy;
#> DROP TABLE #TTAR_era_overlaps;
#> DROP TABLE #TTAR_erafied;
#> DROP TABLE #exc_TTAR_o;
#> DROP TABLE #exc_TTAR_o_to_erafy;
#> DROP TABLE #ex_TTAR_o_overlaps;
#> DROP TABLE #exc_TTAR_o_erafied;
#> DROP TABLE #at_risk_smry_pre_xcl;
#> DROP TABLE #outcome_smry_pre_xcl;
#> DROP TABLE #excluded_person_days;
#> DROP TABLE #excluded_persons;
#> DROP TABLE #excluded_outcomes;
#> DROP TABLE #excl_persons_w_o;
#> DROP TABLE #tscotar_ref;
#> DROP TABLE #incidence_summary;
#>
#> --
#> -- End analysis 0
#> --
#>
#> DROP TABLE #target_ref;
#> DROP TABLE #tar_ref;
#> DROP TABLE #outcome_ref;
#> DROP TABLE #subgroup_ref;
```

4.3 Render SQL with paramaters and execute