Creating a Complex Cohort using Capr

Contents

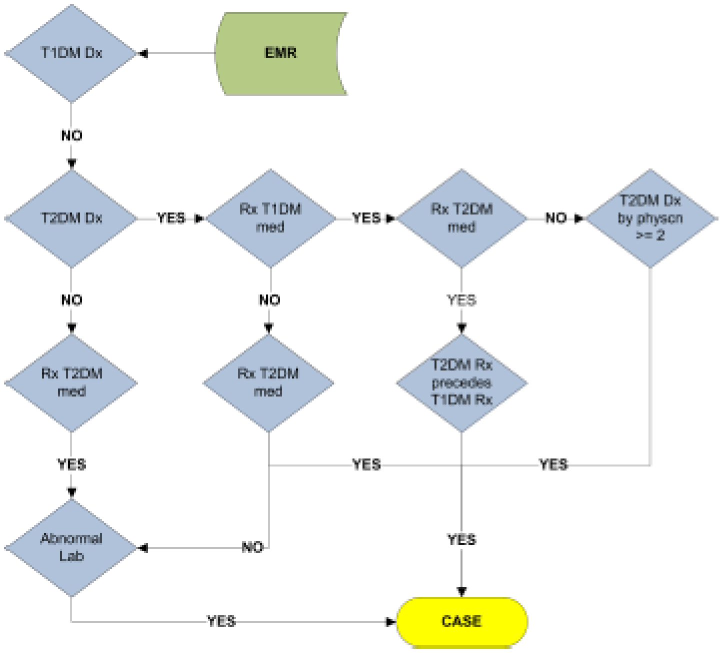
0.1 Introduction

Capr is helpful for cohort definition creation because components of a cohort definition can be split into isolated objects, independent of the full cohort definition. Using Capr, we can now construct smaller pieces of the cohort definition such as queries, counts or groups that can be reincorportated across the cohort definition. The idea of reusable component parts is to construct smaller and simpler objects that can be saved to disk (or in the global environment) to be used as an input to a more complex feature of the cohort definition. In this vignette, we show how we can leverage Capr to create complex cohorts and lay the foundations for more advanced use cases like cohort templating.

The example detailed in this vignette follows follows a complex cohort definition creation example detailed in the following video. Our goal is to create the PheKB type 2 diabetes cohort definition that contains five paths that logically defines a case of type 2 diabetes. In this example we construct smaller pieces of the cohort definition that we can reuse during the construction of the cohort.

The PheKB Type 2 Diabetes Mellitus cohort definition is described by the flowchart:

Figure 1: Algorithm for identifying T2DM cases in the EMR.



To build this cohort we first start with a initial set of presumed cases of T2DM. Patient initially in this cohort must have all of the following: a diagnosis of T2DM (T2Dx), a prescribed medication for T2DM (T2Rx), and presence of an abnormal lab for T2DM. An abnormal lab can be described by any of: ≥ 125 mg/dl for fasting glucose, ≥ 200 mg/dl for random glucose or $\geq 6\%$ for HbA1c. From this initial set we need to restrict initial events to not included persons with a type 1 Diabetes diagnosis. With this a set of presumed cases, we can now confirm T2DM diagnosis using an algorithm composed of 5 pathways based on the above figure:

Path 1: exactly 0 T2Dx, at least 1 T2Rx and at least 1 AbLab

Path 2: at least 1 T2Dx, exactly 0 T1Rx, exactly 0 T2Rx, and at least 1 AbLab

Path 3: at least 1 T2Dx, exactly 0 T1Rx, and at least 1 T2Rx

Path 4: at least 1 T2Dx, at least 1 T1Rx, at least 1T2Rx, and at least 1 occurrence of T2Rx before T1Rx

Path 5: at least 1 T2Dx, at least 1 T1Rx, exactly 0 T2Rx and at least 2 T2Dx

These inclusion rules restrict our initial event, ensuring that we are only recruiting patients with a confirmed T2DM case. In this vignette we outline the process of creating this cohort in Capr.

0.2 Setup

Before we start using Capr, we must load it as a dependent package in R and set up a connection to an OMOP database using `DatabaseConnector`. In order to use Capr's lookup functions, one needs to be connected to an OMOP database with up-to-date vocabulary tables. This helps us find all the concept information, map descendants, and map to the standard concept for cohort definition queries. This setup should be at the top of all Capr scripts to be referenced throughout.

```
library(Capr)
library(DatabaseConnector)
```

```r
#set database connection details
connectionDetails <- createConnectionDetails(
  dbms = "postgresql",
  server = "example.com/datasource",
  user = "me",
  password = "secret",
  port = "5432")

#set connection
connection <- connect(connectionDetails)
#identify vocabulary schema
vocabularyDatabaseSchema <- "vocab"
```

0.3 Building Concept Set Expressions

The first step in cohort creation is to identify medical concepts of interest that capture possible occurrence of type 2 diabetes, for example. Concept set expressions are a fundamental building block to the cohort definition because they group together all codes used to capture a common medical term and establish hierarchical relationships between these sets of codes. As an example, let's say we are interested in all Type 2 Diabetes medications. We want to capture metformin and glipizide and all other medications used prescribed for Type 2 Diabetes. We can group these together as a single concept set. But what about if there are different brands or dosages of these medications, how do we include all of them? For this we need to establish a descendant relationship between the root concept of the ingredient with all its child concepts describing variations on the same ingredient. The concept set expression bundles together codes for all Type 2 Diabetes medications and inherits the descendant concepts used to describe different formulations of the ingredients. We now have a thorough list of all concepts used to describe medications prescribed for Type 2 Diabetes Mellitus. One should refer to documentation on the OMOP vocabularies and concept set creation for more in-depth explanation. Whenever a cohort definition is built, we typically start by identifying the concept set expressions. For the type 2 diabetes cohort we need to identify the following concept set expressions:

1) Diagnosis of Type 2 Diabetes (T2Dx)
2) Medications prescribed for Type 2 Diabetes (T2Rx)
3) Abnormal lab measurements indicating difficulty regulating glucose defined by:

- random glucose $> 200mg/dl$
- fasting glucose $\geq 125mg/dl$
- HbA1c $\geq 6\%$

4) Diagnosis of Type 1 Diabetes (T1Dx)
5) Medications prescribed for Type 1 Diabetes (T1Rx)

Using Capr we start by looking for the concepts required for each of these expressions. Capr allows a user to lookup keywords in the OMOP vocabularies, however for simplicity we pre-identified the concept ids for the example code below. Notice how in the code chunks below we heavily rely on forward piping. The pipe takes the output of the function from the left hand side (lhs) and uses it as the input for a function on the right hand side (rhs). For those familiar with the command line, this is the same logic as piping in bash and something we want to embrace in a functional programming language like R.

```
#Type 2 Diabetes Diagnosis
T2Dx <- getConceptIdDetails(
  conceptIds = 201826,
  connection = connection,
  vocabularyDatabaseSchema = vocabularyDatabaseSchema) %>%
  createConceptSetExpression(
    Name = "Type 2 Diabetes Diagnosis",
    includeDescendants = TRUE)

#Type 2 Diabetes Medications concept IDS
T2RxIds <- c(1502809L, 1502826L, 1503297L, 1510202L,
             1515249L, 1516766L, 1525215L, 1529331L,
             1530014L, 1547504L, 1559684L, 1560171L,
             1580747L, 1583722L, 1594973L, 1597756L)

#create concept set expression
T2Rx <- getConceptIdDetails(
  conceptIds = T2RxIds,
  connection = connection,
   vocabularyDatabaseSchema = vocabularyDatabaseSchema) %>%
  createConceptSetExpression(
    Name = "Type 2 Diabetes Medications",
    includeDescendants = TRUE)

#Type 1 Diabetes Diagnosis
T1Dx <- getConceptIdDetails(
  conceptIds = 201254,
  connection = connection,
  vocabularyDatabaseSchema = vocabularyDatabaseSchema) %>%
  createConceptSetExpression(
    Name = "Type 1 Diabetes Diagnosis",
    includeDescendants = TRUE)
```

Sometimes it is more convenient to use concept codes from a given vocabulary instead of the concept ids. The OMOP vocabularies allow us to look up concepts from both standard and non-standard vocabularies using their original codes. We can map these codes to standard and find the concept ids used for OMOP queries. Below we show code that looks up concept codes and forms the concept set expression used for the cohort definition.

```
#Type 1 Diabetes Medications
T1DRxNormCodes <- paste(c(139825,274783,314684,
                          352385,400008,51428,
                          5856,86009,139953))
T1Rx <- getConceptCodeDetails(
  conceptCode = T1DRxNormCodes,
  vocabulary = "RxNorm",
  connection = connection,
  vocabularyDatabaseSchema = vocabularyDatabaseSchema,
  mapToStandard = TRUE) %>%
  createConceptSetExpression(
    Name = "Type 1 Diabetes Medications",
    includeDescendants = TRUE)

#Abnormal Lab
```

```
AbLabHbA1c <- c("4548-4", "17856-6", "4549-2", "17855-8") %>%
  getConceptCodeDetails(
    conceptCode = ., #places the lhs vector to the rhs of the pipe
    vocabulary = "LOINC",
    connection = connection,
    vocabularyDatabaseSchema = vocabularyDatabaseSchema,
    mapToStandard = TRUE) %>%
  createConceptSetExpression(
    Name = "Abnormal Lab HbA1c",
    includeDescendants = TRUE)

#Ab Lab for Random Glucose (>= 200 mg/dl)
AbLabRandomGluc <- c("2339-0", "2345-7") %>%
  getConceptCodeDetails(
    conceptCode = .,
    vocabulary = "LOINC",
    connection = connection,
    vocabularyDatabaseSchema = vocabularyDatabaseSchema,
    mapToStandard = TRUE) %>%
  createConceptSetExpression(
    Name = "Abnormal Lab Random Glucose",
    includeDescendants = TRUE)

#Ab Lab for Fasting Glucose (>= 125 mg/dl)

AbLabFastingGluc <- c("1558-6") %>%
  getConceptCodeDetails(
    conceptCode = .,
    vocabulary = "LOINC",
    connection = connection,
    vocabularyDatabaseSchema = vocabularyDatabaseSchema,
    mapToStandard = TRUE) %>%
  createConceptSetExpression(
    Name = "Abnormal Lab Fasting Glucose",
    includeDescendants = TRUE)
```

0.4 Building Queries

The next building block of a cohort definition is a query. A query, as defined in the context of circe and Capr, provides the mechanism for identifying an observational event in the CDM. By identifying a domanin we can search for presence of a concept (and its descendants) within the clinical tables of the CDM. For example if we wanted to look for presence of a type 2 diabetes diagnosis we would search the condition occurrence domain for any matches with the Type 2 diabetes concept set expression. In Capr, the signature for creating a query includes the domain from which we wish to query. A concept set expression is typically attached to the query and sometimes can be modified by a list of attributes.

```
#T2Rx Drug Exposure Query
T2RxQuery <- createDrugExposure(conceptSetExpression = T2Rx)


#T1Rx Drug Exposure Query
T1RxQuery <- createDrugExposure(conceptSetExpression = T1Rx)
```

```r

#T2Dx Condition Occurrence Query
T2DxQuery <- createConditionOccurrence(conceptSetExpression = T2Dx)


#T1Dx Condition Occurrence Query
T1DxQuery <- createConditionOccurrence(conceptSetExpression = T1Dx)
```

For the Abnormal labs, there are 3 values that indicate an abnormal level based on the type of measurement. We need to create an attribute with the measurement query for the abnormal lab establishing a threshold for which we can create a binary indicator for presence.

```r
#HbA1c Query with value attribute
AbLabHbA1cQuery <- createMeasurement(
  conceptSetExpression = AbLabHbA1c,
  attributeList = list(
  #add attribute of >= 6% HbA1c
  createValueAsNumberAttribute(Op = "gte", Value = 6)
  ))

#RandomGluc Query with value attribute
AbLabRandomGlucQuery <- createMeasurement(
  conceptSetExpression = AbLabRandomGluc,
  attributeList = list(
  #add attribute of >= 200 mg/dl
  createValueAsNumberAttribute(Op = "gte", Value = 200L)
))

#FastingGluc Query with value attribute
AbLabFastingGlucQuery <- createMeasurement(
  conceptSetExpression = AbLabFastingGluc,
  attributeList = list(
  #add attribute of >= 125 mg/dl
  createValueAsNumberAttribute(Op = "gte", Value = 125L)
))
```

0.5 Creating Counts from Queries

A query looks through a domain and verifies the presence of a concept set expression, modified by the presence of an attribute. What a query lacks is temporal and enumeration logic along a patient timeline. A count component modifies the query to check for the number of occurrences along a temporal spectrum. For example, in our type 2 diabetes pathways we need may need to identify presence of a Type 1 Diabetes medication to confirm the diagnosis. So we need to modify the query for type 1 diabetes medications to ensure that we are counting only those patients who do not have a medication following the initial entry event. Recall the pathways had a series of conditions that all needed to be observed in order for a patient to count within a pathway.

Ex. Path 1: exactly 0 T2Dx, at least 1 T2Rx and at least 1 AbLab

Each item within the pathway is a count. THe count is ensuring we have identified a query by set number of occurrences within a timeline.

For counts in the T2DM phenotype, we set a standard timeline of all days before the initial event and 0 days after the initial event. This means that we want to count events that may have occured anytime before and on the same day, nothing after.

```
#create timeline
tl1 <- createTimeline(StartWindow = createWindow(
  StartDays = "All", StartCoeff = "Before",
  EndDays = 0L, EndCoeff = "After"))
```

Now we can create all the events that take place in this time frame. When we create a count we quantify the occurence of medical events. Either they never happened, they occurred at least once or they occurred at least twice. For each patient, we need to verify that these medical events have been properly observed in order for inclusion. Below we create the sets of counts needed for each pathway.

```
#no occurrence of T1 Diabetes
noT1DxCount <- createCount(Query = T1DxQuery,
                           Logic = "exactly",
                           Count = 0L,
                           Timeline = tl1)

#no occurrence of T2 Diabetes
noT2DxCount <- createCount(Query = T2DxQuery,
                           Logic = "exactly",
                           Count = 0L,
                           Timeline = tl1)


#at least 1 occurrence of T2 Diabetes
atLeast1T2DxCount <- createCount(Query = T2DxQuery,
                                 Logic = "at_least",
                                 Count = 1L,
                                 Timeline = tl1)

#at least 2 occurrence of T2 Diabetes
atLeast2T2DxCount <- createCount(Query = T2DxQuery,
                                 Logic = "at_least",
                                 Count = 2L,
                                 Timeline = tl1)

##################
#Medication Counts
##################


#at least 1 T2DM medication
atLeast1T2RxCount <- createCount(Query = T2RxQuery,
                                 Logic = "at_least",
                                 Count = 1L,
                                 Timeline = tl1)

#no exposure to T2DM medication
noT2RxCount <- createCount(Query = T2RxQuery,
                           Logic = "exactly",
                           Count = 0L,
                           Timeline = tl1)
```

```
#at least 1 T1DM medication
atLeast1T1RxCount <- createCount(Query = T1RxQuery,
                                 Logic = "at_least",
                                 Count = 1L,
                                 Timeline = tl1)

#no exposure to T1DM medication
noT1RxCount <- createCount(Query = T1RxQuery,
                           Logic = "exactly",
                           Count = 0L,
                           Timeline = tl1)

#################
#AbLab Counts
#################

#at least 1 abnormal HbA1c Lab
atLeast1AbLabHbA1cCount <- createCount(Query = AbLabHbA1cQuery,
                                       Logic = "at_least",
                                       Count = 1L,
                                       Timeline = tl1)

#at least 1 abnormal Fasting Glucose Lab
atLeast1AbLabFastingGlucCount <- createCount(Query = AbLabFastingGlucQuery,
                                             Logic = "at_least",
                                             Count = 1L,
                                             Timeline = tl1)

#at least 1 abnormal Random Glucose Lab
atLeast1AbLabRandomGlucCount <- createCount(Query = AbLabRandomGlucQuery,
                                            Logic = "at_least",
                                            Count = 1L,
                                            Timeline = tl1)
```

0.6 Creating the Initial Cohort Entry

With the components created above we can set the initial cohort entry. We defined initial entry as observed occurrence of all of the following events: T2DM diagnosis, prescription of a T2DM medication and the presence of an abnormal lab. In OHDSI the initial recruiting point is called the primary criteria. The primary criteria only searches on queries and is bounded by an observation window distinguishing the minimal amount of continuous observation for the patient to be included.

```
PrimaryCriteria <- createPrimaryCriteria(
  Name = "PC for T2DM Case Phenotype",
  ComponentList = list(T2DxQuery,T2RxQuery,AbLabHbA1cQuery,
                       AbLabFastingGlucQuery,AbLabRandomGlucQuery),
  ObservationWindow = createObservationWindow(
    PriorDays = 0L,
    PostDays = 0L
  ),
  Limit = "All")
```

Recall that we do not want anyone in the cohort to have had a Type 1 Diabetes diagnosis at any time before the primary criteria. We can restrict the initial event through an additional criteria, which defines a group of counts that we wish to exclude from the initial event. An addtional criteria requires a group component input in Capr. Groups will be explained more in the next section.

```
#No T1Dx at any point in patient history
NoT1DxGroup <- createGroup(Name = "No Diagnosis of Type 1 Diabetes",
                           type = "ALL",
                           criteriaList = list(noT1DxCount))

#create additional Criteria
#further restrict the initial capture to people with no T1Dx
AdditionalCriteria <- createAdditionalCriteria(
  Name = "AC for T2DM Case Phenotype",
  Contents = NoT1DxGroup,
  Limit = "First"
)
```

With these two pieces of the cohort we have an initial set of T2DM patients who do no have T1DM. Next we need to confirm the T2DM cases to ensure inclusion into the cohort. In the next section we create the confirmatory pathways.

0.7 Creating T2D Pathways as Groups

In the previous section we alluded to the idea of a group component. Groups are a way of providing boolean logic to counts of events found by queries. In this example, we can bundle multiple counts together as a pathway. Another example of a group before getting to the pathways is the abnormal lab. In our example we don't need the patient to have all the labs, just 1 abnormal lab of the set of three. From the abnormal lab counts created before, we only need 1 to be true for the person to be included in the cohort.

```
atLeast1AbLabGroup <- createGroup(
  Name = "Abnormal labs for HbA1c, Fasting+Random Glucose",
  type = "ANY",
  criteriaList = list(
      atLeast1AbLabHbA1cCount,
      atLeast1AbLabFastingGlucCount,
      atLeast1AbLabRandomGlucCount)
  )
```

Now that we have established all our pieces, we can build each pathway. For a patient to have a confirmed T2DM case they must have all conditions within a single pathway. For pathway 1 this would be no T2Dx, at least 1 T2Rx and at least 1 AbLab prior to the initial event.

```
#Path 1: 0 T2Dx, 1+ T2Rx and 1+ AbLab
Pathway1T2DMGroup <- createGroup(
  Name = "Pathway1",
  Description = "0 T2Dx, 1+ T2Rx and 1+ AbLab",
  type = "ALL",
  criteriaList = list(noT2DxCount, atLeast1T2RxCount),
  Groups = list(atLeast1AbLabGroup))

#Path 2: 1+ T2Dx, 0 T1Rx, 0 T2Rx, and 1+ AbLab
```

```
Pathway2T2DMGroup <- createGroup(
  Name = "Pathway2",
  Description = "1+ T2Dx, 0 T1Rx, 0 T2Rx, and 1+ AbLab",
  type = "ALL",
  criteriaList = list(atLeast1T2DxCount, noT1RxCount, noT2RxCount),
  Groups = list(atLeast1AbLabGroup))

#Path 3: 1+ T2Dx, 0 T1Rx, and 1+ T2Rx
Pathway3T2DMGroup <- createGroup(
  Name = "Pathway3",
  Description = "1+ T2Dx, 0 T1Rx, and 1+ T2Rx",
  type = "ALL",
  criteriaList = list(atLeast1T2DxCount, noT1RxCount, atLeast1T2RxCount)
)

#Path 5: 1+ T2Dx, 1+ T1Rx, 0 T2Rx and 2+ T2Dx
Pathway5T2DMGroup <- createGroup(
  Name = "Pathway5",
  Description = "1+ T2Dx, 1+ T1Rx, 0 T2Rx and 2+ T2Dx",
  type = "ALL",
  criteriaList = list(atLeast1T2DxCount, atLeast1T1RxCount,
                      noT2RxCount, atLeast2T2DxCount)
)
```

Notice we omitted pathway 4. This is because pathway 4 has a special nested structure: a T2Rx must occur before a T1Rx. To create this nesting within T1Rx query, we require a special correlated criteria attribute. This attribute nests a group within a query, meaning that the query is only valid if the nested group is also true. In this case we want the timeline to be All days before and 1 day before, so that we are only counting T2Rx occuring before the T1Rx. The code chunk below shows how this nested criteria is created:

```
tl2 <- createTimeline(StartWindow = createWindow(
  StartDays = "All", StartCoeff = "Before",
  EndDays = 1L, EndCoeff = "Before"))

PriorT2RxCount <- createCount(
  Query = T2RxQuery,
  Logic = "at_least",
  Count = 1L,
  Timeline = tl2
)

PriorT2RxNestedGroup <- createCorrelatedCriteriaAttribute(
  createGroup(
    Name = "Nested Group T2Rx before T1Rx",
    type = "ALL",
    criteriaList = list(PriorT2RxCount)
  )
)

T2RxBeforeT1RxCount <- createDrugExposure(
  conceptSetExpression = T1Rx,
  attributeList = list(PriorT2RxNestedGroup)) %>%
  createCount(Logic = "at_least", Count = 1L,
              Timeline = tl1)
```

With this nested count in place we can build the 4th pathway, which requires at least 1 T2Dx, at least 1 T1Rx, at least 1 T2Rx, and the T2Rx to occur before the T1Rx.

```
#Path 4: 1+ T2Dx, 1+ T1Rx, 1+T2Rx, and 1+ T2Rx < T1Rx
Pathway4T2DMGroup <- createGroup(
  Name = "Pathway4",
  Description = "1+ T2Dx, 1+ T1Rx, 1+T2Rx, and 1+ T2Rx < T1Rx",
  type = "ALL",
  criteriaList = list(atLeast1T2DxCount, atLeast1T1RxCount,
                      T2RxBeforeT1RxCount)
)
```

Now we create the inclusion rules for this cohort. Based on the original entry set, we further restrict to only contain those that follow a set of rules. The patients only need to be in 1 of any of the pathways, thus the boolean for our group component is “any”.

```
#T2DM Case Group
T2DMCase <- createGroup(
  Name = "Case for T2DM using algorithm",
  type = "ANY",
  Groups = list(Pathway1T2DMGroup, Pathway2T2DMGroup, Pathway3T2DMGroup,
                Pathway4T2DMGroup, Pathway5T2DMGroup)
)

#keep T2DM cases if they meet 1 of the 5 pathways
InclusionRules <- createInclusionRules(
  Name = "IRs for T2DM Case Phenotype",
  Contents = list(T2DMCase),
  Limit = "First"
)
```

0.8 Finalizing the Cohort Definition

In the cohort definition we have so far captured entry and restriction, but what about exit. A cohort is defined by an entry an exit so that we understand a patient’s time at risk. In this cohort exit may be defined in two parts: end of continuous observation (meaning the last recorded timestamp available) or a censoring event. If the patient were to be diagnosed with Type 1 Diabetes their events are no longer observed since they can not be a valid T2DM case. We can create a censoring criteria based on a query, shown in the code chunk below:

```
#person exits cohort if there is a diagnosis of T1DM
CensoringCriteria <- createCensoringCriteria(
  Name = "Censor of T1DM cases",
  ComponentList = list(T1DxQuery)
)
```

Now the cohort definition is done, and we can create the full object in Capr as so:

```
T2DMPhenotype <- createCohortDefinition(
  Name = "PheKB T2DM Definition",
  PrimaryCriteria = PrimaryCriteria,
  AdditionalCriteria = AdditionalCriteria,
  InclusionRules = InclusionRules,
  CensoringCriteria = CensoringCriteria
)
```

If we wanted to get the json from the cohort we can do so by compiling the cohort in the circe-be engine. From here we can use `CirceR` and `DatabaseConnector` to run the cohort definition query against the database to generate counts (this step is skipped).

```
T2DMPhenotypeJson <- compileCohortDefinition(T2DMPhenotype)
```

If we wanted to check that this is a valid cohort, we can copy the json output into ATLAS.

```
{
 "ConceptSets": [
 {
 "id": 0,
"name": "Type 2 Diabetes Diagnosis",
"expression": {
 "items": [
 {
 "concept": {
 "CONCEPT_ID": 201826,
"CONCEPT_NAME": "Type 2 diabetes mellitus",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "44054006",
"DOMAIN_ID": "Condition",
"VOCABULARY_ID": "SNOMED",
"CONCEPT_CLASS_ID": "Clinical Finding"
},
"includeDescendants": true
}
]
}
},
{
 "id": 1,
"name": "Type 2 Diabetes Medications",
"expression": {
 "items": [
 {
 "concept": {
 "CONCEPT_ID": 1594973,
"CONCEPT_NAME": "chlorpropamide",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "2404",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1515249,
"CONCEPT_NAME": "troglitazone",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "72610",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1580747,
"CONCEPT_NAME": "sitagliptin",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "593411",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1597756,
"CONCEPT_NAME": "glimepiride",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "25789",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1525215,
"CONCEPT_NAME": "pioglitazone",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "33738",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1510202,
"CONCEPT_NAME": "miglitol",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "30009",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1530014,
"CONCEPT_NAME": "acetohexamide",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "173",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1502826,
"CONCEPT_NAME": "nateglinide",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "274332",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1503297,
"CONCEPT_NAME": "metformin",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "6809",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1502809,
"CONCEPT_NAME": "tolazamide",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "10633",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1560171,
"CONCEPT_NAME": "glipizide",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "4821",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1583722,
"CONCEPT_NAME": "exenatide",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "60548",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1529331,
"CONCEPT_NAME": "acarbose",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "16681",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1547504,
"CONCEPT_NAME": "rosiglitazone",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "84108",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1559684,
"CONCEPT_NAME": "glyburide",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "4815",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1516766,
"CONCEPT_NAME": "repaglinide",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "73044",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
}
]
}
},
{
 "id": 2,
"name": "Abnormal Lab HbA1c",
"expression": {
 "items": [
 {
 "concept": {
 "CONCEPT_ID": 3004410,
"CONCEPT_NAME": "Hemoglobin A1c/Hemoglobin.total in Blood",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "4548-4",
"DOMAIN_ID": "Measurement",
"VOCABULARY_ID": "LOINC",
"CONCEPT_CLASS_ID": "Lab Test"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 3003309,
"CONCEPT_NAME": "Hemoglobin A1c/Hemoglobin.total in Blood by Electrophoresis",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "4549-2",
"DOMAIN_ID": "Measurement",
"VOCABULARY_ID": "LOINC",
"CONCEPT_CLASS_ID": "Lab Test"
},
"includeDescendants": true
```

```
},
{
 "concept": {
 "CONCEPT_ID": 3007263,
"CONCEPT_NAME": "Hemoglobin A1c/Hemoglobin.total in Blood by calculation",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "17855-8",
"DOMAIN_ID": "Measurement",
"VOCABULARY_ID": "LOINC",
"CONCEPT_CLASS_ID": "Lab Test"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 3005673,
"CONCEPT_NAME": "Hemoglobin A1c/Hemoglobin.total in Blood by HPLC",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "17856-6",
"DOMAIN_ID": "Measurement",
"VOCABULARY_ID": "LOINC",
"CONCEPT_CLASS_ID": "Lab Test"
},
"includeDescendants": true
}
]
}
},
{
 "id": 3,
"name": "Abnormal Lab Fasting Glucose",
"expression": {
 "items": [
 {
 "concept": {
 "CONCEPT_ID": 3037110,
"CONCEPT_NAME": "Fasting glucose [Mass/volume] in Serum or Plasma",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "1558-6",
"DOMAIN_ID": "Measurement",
"VOCABULARY_ID": "LOINC",
"CONCEPT_CLASS_ID": "Lab Test"
},
"includeDescendants": true
}
```

```
]
}
},
{
 "id": 4,
"name": "Abnormal Lab Random Glucose",
"expression": {
 "items": [
 {
 "concept": {
 "CONCEPT_ID": 3000483,
"CONCEPT_NAME": "Glucose [Mass/volume] in Blood",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "2339-0",
"DOMAIN_ID": "Measurement",
"VOCABULARY_ID": "LOINC",
"CONCEPT_CLASS_ID": "Lab Test"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 3004501,
"CONCEPT_NAME": "Glucose [Mass/volume] in Serum or Plasma",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "2345-7",
"DOMAIN_ID": "Measurement",
"VOCABULARY_ID": "LOINC",
"CONCEPT_CLASS_ID": "Lab Test"
},
"includeDescendants": true
}
]
}
},
{
 "id": 5,
"name": "Type 1 Diabetes Diagnosis",
"expression": {
 "items": [
 {
 "concept": {
 "CONCEPT_ID": 201254,
"CONCEPT_NAME": "Type 1 diabetes mellitus",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "46635009",
"DOMAIN_ID": "Condition",
"VOCABULARY_ID": "SNOMED",
"CONCEPT_CLASS_ID": "Clinical Finding"
},
"includeDescendants": true
}
]
}
},
{
 "id": 6,
"name": "Type 1 Diabetes Medications",
"expression": {
 "items": [
 {
 "concept": {
 "CONCEPT_ID": 1516976,
"CONCEPT_NAME": "insulin detemir",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "139825",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1550023,
"CONCEPT_NAME": "insulin lispro",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "86009",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1513876,
"CONCEPT_NAME": "insulin lispro protamine, human",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "314684",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1544838,
"CONCEPT_NAME": "insulin glulisine, human",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "400008",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1502905,
"CONCEPT_NAME": "insulin glargine",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "274783",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1517998,
"CONCEPT_NAME": "pramlintide",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "139953",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1531601,
"CONCEPT_NAME": "insulin aspart protamine, human",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "352385",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
},
{
 "concept": {
 "CONCEPT_ID": 1567198,
"CONCEPT_NAME": "insulin aspart, human",
"STANDARD_CONCEPT": "S",
"STANDARD_CONCEPT_CAPTION": "Standard",
"INVALID_REASON": "V",
"INVALID_REASON_CAPTION": "Valid",
"CONCEPT_CODE": "51428",
"DOMAIN_ID": "Drug",
"VOCABULARY_ID": "RxNorm",
"CONCEPT_CLASS_ID": "Ingredient"
},
"includeDescendants": true
}
]
}
}
],
"PrimaryCriteria": {
 "CriteriaList": [
 {
 "ConditionOccurrence": {
 "CodesetId": 0
}
},
{
 "DrugExposure": {
 "CodesetId": 1
}
},
{
 "Measurement": {
 "CodesetId": 2,
"ValueAsNumber": {
 "Value": 6,
"Op": "gte"
}
}
},
{
 "Measurement": {
```

```
 "CodesetId": 3,
"ValueAsNumber": {
 "Value": 125,
"Op": "gte"
}
}
},
{
 "Measurement": {
 "CodesetId": 4,
"ValueAsNumber": {
 "Value": 200,
"Op": "gte"
}
}
}
],
"ObservationWindow": {
 "PriorDays": 0,
"PostDays": 0
},
"PrimaryCriteriaLimit": {
 "Type": "All"
}
},
"AdditionalCriteria": {
 "Type": "ALL",
"CriteriaList": [
 {
 "Criteria": {
 "ConditionOccurrence": {
 "CodesetId": 5
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 0,
"Count": 0
}
}
],
"DemographicCriteriaList": [],
"Groups": []
},
"QualifiedLimit": {
 "Type": "First"
},
"ExpressionLimit": {
 "Type": "First"
},
"InclusionRules": [
 {
 "name": "Case for T2DM using algorithm",
"expression": {
 "Type": "ANY",
"CriteriaList": [],
"DemographicCriteriaList": [],
"Groups": [
 {
 "Type": "ALL",
"CriteriaList": [
 {
 "Criteria": {
 "ConditionOccurrence": {
 "CodesetId": 0
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 0,
"Count": 0
}
},
{
 "Criteria": {
 "DrugExposure": {
 "CodesetId": 1
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
}
],
"DemographicCriteriaList": [],
"Groups": [
 {
 "Type": "ANY",
"CriteriaList": [
 {
 "Criteria": {
 "Measurement": {
 "CodesetId": 2,
"ValueAsNumber": {
 "Value": 6,
"Op": "gte"
}
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
},
{
 "Criteria": {
 "Measurement": {
 "CodesetId": 3,
"ValueAsNumber": {
 "Value": 125,
"Op": "gte"
}
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
},
{
 "Criteria": {
 "Measurement": {
 "CodesetId": 4,
"ValueAsNumber": {
 "Value": 200,
"Op": "gte"
}
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
}
],
"DemographicCriteriaList": [],
"Groups": []
}
]
},
{
 "Type": "ALL",
"CriteriaList": [
 {
 "Criteria": {
 "ConditionOccurrence": {
 "CodesetId": 0
}
},
"StartWindow": {
 "Start": {
```

```
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
},
{
 "Criteria": {
 "DrugExposure": {
 "CodesetId": 6
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 0,
"Count": 0
}
},
{
 "Criteria": {
 "DrugExposure": {
 "CodesetId": 1
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 0,
"Count": 0
}
}
],
"DemographicCriteriaList": [],
"Groups": [
 {
 "Type": "ANY",
"CriteriaList": [
 {
 "Criteria": {
 "Measurement": {
 "CodesetId": 2,
"ValueAsNumber": {
 "Value": 6,
"Op": "gte"
}
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
},
{
 "Criteria": {
 "Measurement": {
 "CodesetId": 3,
"ValueAsNumber": {
 "Value": 125,
"Op": "gte"
}
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
},
{
 "Criteria": {
 "Measurement": {
 "CodesetId": 4,
"ValueAsNumber": {
 "Value": 200,
"Op": "gte"
}
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
}
],
"DemographicCriteriaList": [],
"Groups": []
}
]
},
{
 "Type": "ALL",
"CriteriaList": [
 {
 "Criteria": {
 "ConditionOccurrence": {
 "CodesetId": 0
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
},
{
 "Criteria": {
 "DrugExposure": {
 "CodesetId": 6
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 0,
"Count": 0
}
},
{
 "Criteria": {
 "DrugExposure": {
 "CodesetId": 1
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
```

```
}
],
"DemographicCriteriaList": [],
"Groups": []
},
{
 "Type": "ALL",
"CriteriaList": [
 {
 "Criteria": {
 "ConditionOccurrence": {
 "CodesetId": 0
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
},
{
 "Criteria": {
 "DrugExposure": {
 "CodesetId": 6
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
},
{
 "Criteria": {
```

```
 "DrugExposure": {
 "CodesetId": 6,
"CorrelatedCriteria": {
 "Type": "ALL",
"CriteriaList": [
 {
 "Criteria": {
 "DrugExposure": {
 "CodesetId": 1
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 1,
"Coeff": -1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
}
],
"DemographicCriteriaList": [],
"Groups": []
}
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
}
],
"DemographicCriteriaList": [],
"Groups": []
},
{
 "Type": "ALL",
"CriteriaList": [
 {
 "Criteria": {
 "ConditionOccurrence": {
 "CodesetId": 0
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
},
{
 "Criteria": {
 "DrugExposure": {
 "CodesetId": 6
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 1
}
},
{
 "Criteria": {
 "DrugExposure": {
 "CodesetId": 1
}
},
"StartWindow": {
 "Start": {
```

```
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 0,
"Count": 0
}
},
{
 "Criteria": {
 "ConditionOccurrence": {
 "CodesetId": 0
}
},
"StartWindow": {
 "Start": {
 "Coeff": -1
},
"End": {
 "Days": 0,
"Coeff": 1
},
"UseEventEnd": false,
"UseIndexEnd": false
},
"Occurrence": {
 "Type": 2,
"Count": 2
}
}
],
"DemographicCriteriaList": [],
"Groups": []
}
]
}
}
],
"CensoringCriteria": [
 {
 "ConditionOccurrence": {
 "CodesetId": 5
}
}
],
"CollapseSettings": {
 "CollapseType": "ERA",
"EraPad": 0
```

```
},
"CensorWindow": {},
"cdmVersionRange": ">=5.0.0"
}
```